creative computing

Nov-Dec 1977
vol 3, no 6

Austria Sch 32.00
Belgium BF 70.00
Canada $2.00
Denmark Kr 11.00
Finland Mk 7.00
France F 9.00
Germany DM 4.50
Great Britain £ 1.00
Greece Dr. 65.00
Holland Dfl 5.00
Italy L 1500
Norway Kr 10.00
Portugal Esc 55.00
Spain Ptas 120.00
Sweden Kr 8.00
Switzerland SF 4.50
USA $1.50

the magazine of recreational and educational computing

$1.50

Gamboling in Atlantic City

Evaluation of Three 8080 8K BASICs

Let the Computer Write Those Final Exams

S-100-Compatible Boards and Kits

Computer Games:
- Mastermind II
- Othello

Shoestring Timeshare

Programming Techniques: File Structures

Smart Electronic Games

You can now have the industry's finest microcomputer with that all-important disk drive

The professional-grade microcomputer for professionals

Shown with optional bench cabinet

YOU CAN GET THAT ALL-IMPORTANT SOFTWARE, TOO

Loading your programs and files will take you only a few seconds with the new Cromemco Z-2D computer.

You can load fast because the Z-2D comes equipped with a 5″ floppy disk drive and controller. Each diskette will store up to 92 kilobytes.

Diskettes will also store your programs inexpensively—much more so than with ROMs. And ever so much more conveniently than with cassettes or paper tape.

The Z-2D itself is our fast, rugged, professional-grade Z-2 computer equipped with disk drive and controller. You can get the Z-2D with either single or dual drives (dual shown in photo).

CROMEMCO HAS THE SOFTWARE

You can rely on this: Cromemco is committed to supplying quality software support.

For example, here's what's now available for our Z-2D users:

CROMEMCO FORTRAN IV COMPILER: a well-developed and powerful FORTRAN that's ideal for scientific use. Produces optimized, relocatable Z-80 object code.

CROMEMCO 16K DISK BASIC: a powerful pre-compiling interpreter with 14-digit precision and powerful I/O handling capabilities. Particularly suited to business applications.

CROMEMCO Z-80 ASSEMBLER: a macro-assembler that produces relocatable object code. Uses standard Z-80 mnemonics.

ADVANCED CONTROLLER CARD

The new Z-2D is a professional system that gives you professional performance.

In the Z-2D you get our well-known 4-MHz CPU card, the proven Z-2 chassis with 21-slot motherboard and 30-amp power supply that can handle 21 cards and dual floppy drives with ease.

Then there's our new disk controller card with special features:

- Capability to handle up to 4 disk drives
- A disk bootstrap Monitor in a 1K 2708 PROM
- An RS-232 serial interface for interfacing your CRT terminal or teletype
- LSI disk controller circuitry

Z-2 USERS:
Your Z-2 was designed with the future in mind. It can be easily retrofitted with everything needed to convert to a Z-2D. Only $935 kit; or $1135 for assembled retrofit package.

We're able to put all of this including a UART for the CRT interface on just one card because we've taken the forward step of using LSI controller circuitry.

STORE/FACTORY

Contact your computer store or Cromemco factory now about the Z-2D. It's a real workhorse that you can put to professional or OEM use now.

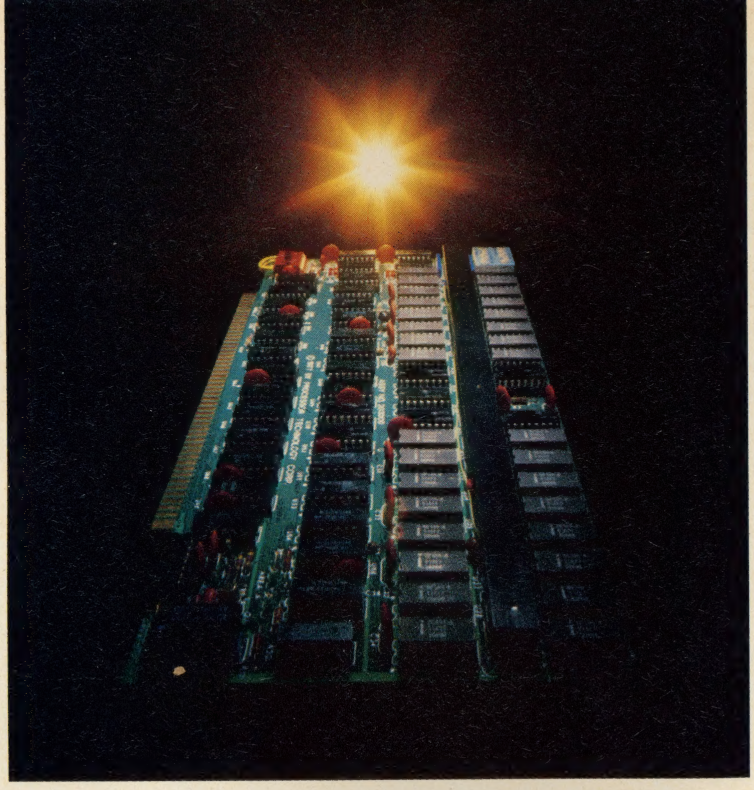
Kit: Z-2D with 1 disk drive (Model Z2D-K) .... $1495.
Assembled: Z-2D fully assembled and tested (Model Z2D-W) .... $2095.
Additional disk drive (Model Z2D-FDD) .... $495.

SOFTWARE

(On standard IBM-format soft-sectored mini diskettes)

16K BASIC (Model FDB-S) $95
FORTRAN IV (Model FDF-S) $95
Z-80 Assembler (Model FDA-S) $95

Cromemco
incorporated
Specialists in computers and peripherals
2400 CHARLESTON RD., MOUNTAIN VIEW, CA 94043 • (415) 964-7400



in this issue . . .

articles

fiction & foolishness

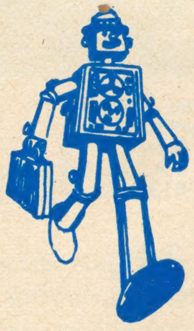
The Cover

The cover is an original drawing by Paul Stinson, a noted science-fiction artist, whose works have been displayed at many galleries. This versatile artist also created the highly original March-April cover for us.

reviews & resources

things to do-games

departments



Nov-Dec 1977 Volume 3, Number 6 Consecutive Issue No. 18

Creative Computing magazine is published bi-monthly by Creative Computing, P.O. Box 789-M, Morristown, NJ 07960. (Editorial office: 51 Dumont Place, Morristown, NJ 07960. Phone: (201) 540-0445.)

Domestic Subscriptions: 1-year $8, 2-year $15, 3-year $21. Subscription orders, change of address, P.O. Form 3579 to Creative Computing, P.O. Box 789-M, Morristown, NJ 07960. Call 800-631-8112 toll-free to order a subscription (to be charged only to a bank card).

Foreign Subscriptions:
Great Britain. 1-year £ 6.25, 2-year £ 11.90, 3-year £ 16.90. Orders to Creative Computing, 60 Porchester Road, Southampton S02 7JD, England.
Europe, 1-year subscription. Austria Sch 180, Belgium BF 4000, Denmark Kr65, Finland Mr40, France Fr50, Germany DM25, GreeceDr360, Holland Dfl27, Italy L8500, Norway Kr55, Portugal Esc320, Spain Ptas700. Sweden Kr45, Switzerland SF 25. Orders to Pan Atlantic Computer Systems GmbH, Frankfurter Strasse 78, D61 Darmstadt, German Fed Rep.
Other Countries 1-year $12, 2-year $23, 3-year $33 (surface postage, U.S. dollars). Orders to Creative Computing, P.O. Box 789-M, Morristown, NJ 07960.

Second class postage paid at Morristown, New Jersey and at additional mailing offices.

Publisher
David H. Ahl

Editor-in-Chief
Stephen B. Gray

Managing Editor
Burchenal Green

Associate Editor
Steve North

Software & Reviews Editor
John Lees

Higher Education Editors
Merl Miller
A. Kent Morton

Technology Editors
Thomas R. Kibler
Alan Salisbury

Careers Editor
Eleanor Corr

Humor Editor
Jack Ludwig

Art Director
Elyse J. Fox

Marketing Manager
Linda Harrison

Business Manager
Mary M. Williams

Retail Marketing Rep.
Linda Eckerstrom

Book Service
Eric Sellin
Mark Kuschke

Customer Service
Jeanne Tick

Subscriptions
Nancy Herbst
Carol Cassata

Advertising Sales

Western States, Texas
Jules E. Thompson
Hearst Building, Suite IIII
5 Third Street
San Francisco, CA 94103
(415) 362-8361

Bert Charlton
2560 Via Tejon
Palos Verdes Estates, CA 90274
(213) 378-8361

Mid-Atlantic, Northeast
Charles Lynch
36 Sohier Street
Cohasset, MA 02025
(617) 383-6136

Elsewhere
Burchenal Green
(914) 739-1144 or
(201) 540-0445

MEMBER
COSMEP
COMMITTEE OF SMALL MAGAZINE EDITORS AND PUBLISHERS

...notices...

ACM Special-Interest Group for Personal Computing

Having established the need for a technical organization in the area of personal computing, SIGPC has been chartered and will concern itself with the design and application of computer systems for personal uses. This includes personal computer systems for home, clerical, small-business management, and recreational uses. It also includes the technology of such systems in software and hardware, and emphasizes techniques appropriate to the integration of such tools as graphics, speech, data management, and music systems.

Dues include a subscription to the newsletter, and are $5 for ACM Members or associate members or student members; $13 for non-ACM member of SIGPC (open only to individuals whose "major professional allegiance is in a field other than information processing"). A subscription to the newsletter only is $12.

ACM, 1133 Ave. of the Americas, New York, NY 10036, Attn: SIG/SIC.

Second West Coast Computer Faire

The second West Coast Computer Faire is scheduled for March 3-5, 1978, in the brand-new San Jose Convention Center in San Jose, California, which, as its promoters tell us, is at the southern end of the San Francisco Peninsula, and is in the *middle* of "Silicon Valley" — center of the semiconductor industry.

Projections are that 10,000 to 15,000 people will attend, that there will be 150 to 190 exhibitors, and 50 to 100 speakers.

One of the rules at the First Computer Faire was that manufacturers should not present talks concerning their products. The promotors have since concluded that this was an error, "a disservice to both the attendees and the manufacturers." Thus manufacturers are now being encouraged to prepare tutorials concerning their products, "comprehensive and educational in nature," but not heavy sales pitches.

Deadline for papers is January 2, 1978.

Jim Warren, The Computer Faire, P.O. Box 1579, Palo Alto, CA 94302. (415) 851-7664.

Personal & Small Business Computer Expo

To "provide the small-computer industry with an annual forum and meeting place to pace the progress of this fast-growing and important segment of the data-processing industry," the Personal & Small Business Computer Expo will be held on January 13-15, 1978, at the Washington Hilton Hotel in Washington, DC.

This exposition for the personal and small business computer industry will be open to the trade, small-business managers, hobbyists, and all other small-computer users in government, education and industry.

E.F. Felsburg, Executive Director, Felsburg Associates, Inc., P.O. Box 624, 9430 Lanham-Severn Road, Seabrook, MD 20801. (301) 459-1590.

Black Box Wanted

Creative Computing will pay $25.00 for the best simulation of the game of Black Box received by March 1, 1978. Must be in a "standard" dialect of BASIC for play on a hard-copy terminal. An alternative version for a VDM, TV Dazzler or other display device would be nice too. Clean listing and sample run on white paper, please. Wrap paper tape, if included, in plastic wrap or foil to prevent oil seepage. Please don't write us for the rules to Black Box; it's available from retailers of adult games.

Notice to Program Writers

We are unable to accept *typed* programs for publication because we have no way of knowing if they are correct.

When we receive a computer-output LISTing, we assume the accompanying RUN was generated from the LIST, which provides proof that the program has been debugged.

Sorry to make problems for readers who have no printer or tape punch, but we just don't have the time to check out typed programs by making tapes of them.


System 8813
PolyMorphic Systems
Load

Reflections At The End Of Our Third Year

From time to time I get questions from people at conventions, and from readers through the mail, as well as from my own staff, as to what the philosophy of *Creative Computing* is. Where are we going? What's the magazine trying to be? And as we enter our fourth year, I suppose it's an approporate time for the publisher to sit back and speculate a little bit on where this thing is going, this monster I created about four years ago, that in some ways has gotten completely out of hand.

Our direction over the first three years has obviously changed. This is probably most apparent when you consider two major groups of readers, educators and home-computer users. Some educators have complained that we have given up the education market and we have become hobbyist-oriented. To this I would say, nicely but bluntly, that (1) you educators are rather narrow-minded if you think that the types of things *Creative Computing* is running are not appropriate for education just because they happen to focus increasingly on microcomputers and on manufacturers who don't have the vast dealer networks or army of salesmen to be calling on schools. The educational value of a computer may be far greater if the student is dealing with an accessible, hands-on micro than if he's dealing with a computer that can't be touched by anyone but a "qualified field-service technician."

And (2) I would ask the skeptical educator: aren't you happy to have other people reading a thoughtful and pedagogically sound magazine? Maybe *Creative* doesn't look like a typical educational magazine anymore because we're not running the dry scholarly articles that "should" appear in an educational magazine. However, we are running ideas that have a sound basis but we're presenting them in easy-to-understand terms. This is rarely found in educational magazines which somehow feel that to put things in four-syllable words makes them more acceptable to faculty members who probably don't have more understanding of these fancy words than the student, and certainly aren't any more interested or inclined to read them.

Actually we have very, very few educators who are criticizing *Creative Computing* and I hail the foresight of the majority of our educational readers.

But that doesn't get at the question of where we're going and what our philosophical basis is. We have many diverse groups of readers: educational computer users, hobbyist users, time-sharing users, users of minis or micros, and big-system users. I guess the one common denominator in all of those descriptions is the word *user.* The assumption that we make at *Creative Computing* is that our readers are interesting in *using* their computer power. They are not tinkerers. They are not electronic nuts. Oh sure, they may well tinker around to get their system working. They may well be immersed into electronics, but the main object of their purchasing a computer in the first place is not to solder boards together forever, or to learn about various communications protocols, or to build a fancy woodgrain cabinet for their finished machine, but it is to use it for some application. Be that personal, or educational, or for home-management, or building management, or for small business. The point is that the readers of *Creative Computing,* we believe, are interested in applications.

Now, there are two levels to the applications. Some applications are original and unique enough to a particular reader that he or she is the only person that can or will produce it. For these readers, we feel that our obligation is to provide programming techniques to make writing their application easier and more efficient. Techniques like sort routines, large-number processing, recursion, shuffling routines, and data-based management schemes. Also for the reader who is writing his own application, we present materials on different computer languages. Perhaps one application could be better implemented in APL, or Fortran, or perhaps the version of BASIC that is available to the reader is not compatible with that particular application because it doesn't have the proper string-handling or file-handling capability. Consequently, we feel that another one of our jobs is to take a look at different types of BASIC compilers and their strengths and weaknesses. Ditto other high-level languages.

In addition, for the reader who is writing his own application, we feel that it's worthwhile to present articles like "Computational Unsolvability," or "Catastrophe Theory." The first discussed certain types of problems which are not amenable to computational solution. The second looked at some completely unexpected types of curves that provide a framework for analyzing discontinuous events in fields such as biology, sociology or even the stock market.

For another group of readers who are not so much interested in writing their own unique application, but rather in using finished applications or perhaps combining several to do something of use, we present a certain number of complete programs. So far, most of these have been in the area of games, puzzle solving, and various other recreational applications. In the future, we expect these applications to broaden out into more serious areas such as text editing, word processing, building management, household management, and business functions. Of course, we'll continue to present games and recreational applications since these have proved extremely popular in the past and, by my best estimate, at least 70% of the home users are using their computers almost exclusively for games and leisure.

Along with applications, I think it's equally important for one to consider what is really worthwhile doing with a computer. There are lots of things that *can* be done, but what is really *worth* doing? Now there is no hard-and-fast,

cut-and-dried answer to that question. Every person is going to have to decide on his or her own part what is worth doing with his (or her) computer. However, from time to time, we will present articles and even fiction and poetry which will speculate on future computer applications and where they might lead. Of course, we would hope that any fiction or poetry that we would put in the magazine would be interesting and entertaining in its own right, but if we choose our material carefully, maybe we can also provide a message in this type of material.

Since we feel that our readers have a user orientation, one other mission that we feel is appropriate is to present accurate reviews of books and other materials aimed at a user. As a result, we are the only publication going into the hobbyist market that has any kind of serious book-review section. If you read these reviews, you'll realize that many of them go far beyond reviewing the specific book in question, but present ideas, commentary, and thoughts of the reviewer. Our reviews, in other words, are considerably closer in character to those appearing in the *New York Times Book Review* than to the *ACM Computing Reviews*.

I think it's interesting to note that our overlap in readership with other magazines is heaviest with *Scientific American* by a rather wide margin and then, secondly, with *Byte* magazine. What this indicates to me is that we have a very well-educated reader and one with a rather broad set of interests. Of course, they are interested in the hardware; many of them who are not in an environment with ready-made computers and terminals at their disposal are trying to construct their own systems and consequently have the need of a hardware-oriented magazine such as *Byte*. However, I think they realize that eventually their system is going to get up and running, or perhaps they *hope* it is, and at that point, then they're going to need an applications-oriented magazine like *Creative Computing*.

Some months ago, we had a major debate among our staff members and advertising reps whether or not *Creative Computing* should become a monthly magazine. There were good arguments on both sides. However, what it finally boiled down to was the fact that we were making a small amount of money and we had to choose how to invest that money most wisely. If we had become a monthly magazine, we would have had to put considerably more resources into soliciting advertising, since very few magazines can survive on subscription revenue alone. In other words, all of our excess capital for at least the next nine to twelve months would have had to go into catering to, and soliciting, advertising. Alternatively, we could put the same money into improving the editorial quality and hiring additional people on the editorial side of the magazine. This is what I elected to do. *Creative Computing* now has a larger editorial staff of full-time writers and editors than any other educational or hobbyist computer magazine, bar none. For the foreseeable future, therefore, my intent is to improve upon the editorial side of *Creative Computing*. My intent is for it to be of the absolute highest quality. That is not to say there will not be occasional typos, or errors in program listings or other gliches. However, we have elected to make *Creative Computing* a leader as a result of its editorial quality rather than its advertising content. My belief is that, eventually advertisers will discover that *Creative Computing* readers like the magazine for its editorial quality and therefore will advertise with us and one will follow the other. On the other hand, its a risky business going out and soliciting advertising, hoping that you get it, and then when you do, hiring the editorial people to beef up the quality of the magazine. One reader termed us the *Scientific American* of the personal computing field. That's a very complimentary, but I think very apt, description of *Creative Computing*.

Where do we go from here? Well, certainly more applications in a broader set of areas, more programming techniques, more speculation on what should be done with computing power, more reviews, and, oh sure, some more fiction and foolishness too, because along with all of the pedagogically sound serious material, there is a streak of foolishness in all of us. Or if not in all of us, there at least is in me, and as long as I'm the publisher, you're going to continue to get some cartoons and triviality along with the serious stuff.

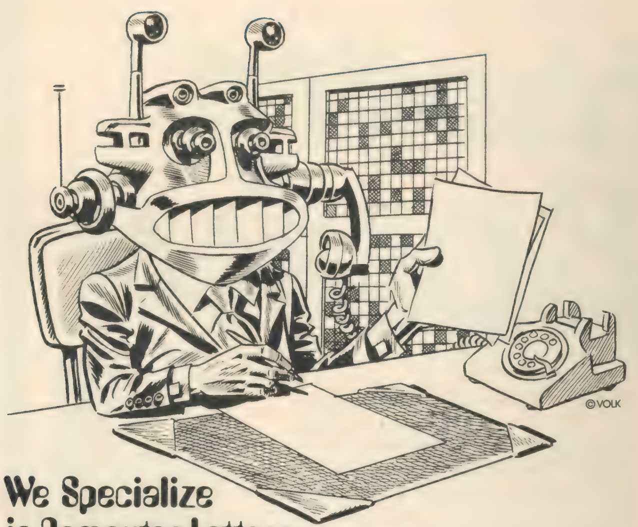
But we'll be evolving too. We'll be evolving in response to technological advances and also in response to changing desires of our readership. The only way we know about these changing desires is from your cards and letters, so keep them coming. Oh, we also know from subscription cancellations and I'd just as soon you *not* keep them coming. Actually, we have an exceptionally high renewal rate compared to the magazine industry at large, which pleases me greatly. However, we're not perfect. I recognize that and I'm happy for any comments that will bring us a little bit closer to what you, the readers, want to see in *Creative Computing*.

It goes without saying that articles along the lines mentioned above, or dealing with new things that you think would be suitable for *Creative Computing*, are always welcome. We pay for articles; you'll never get rich and you won't be able to retire on the money. On the other hand, it will probably compensate you for the time of sitting down and typing out your ideas in a coherent fashion to share with other readers. But one way or another, as an author or as a commentator or as a critic, let us hear from you.

Over the last three years, our subscription solicitation letters have frequently invited people to join the *Creative Computing* family. I still think of the entire readership of *Creative Computing* in that light, in their many and varied roles, as being part of one large family, one large family of intelligent, wise computer users. I hope you do too. ■

David H. Ahl

put...input/output...in



We Specialize in Computer Letters

In Time for Christmas

Dear Editor:

The July-August issue of *Creative Computing* contained both your elegant statistical readership survey and a story about my computer T-shirts, in which I called for suggestions for new phrases to print on them. I've never received so many delightful letters! I thought you might be interested in some unstatistical data on the responses I got. Your 91% reader interest in humor is confirmed. Here are some samples: BUBBLE LOGIC, THINKERTOY, GARBAGE IN — GOSPEL OUT, TODAY IS A COMPUTER SIMULATION, OPERATOR ERORR, TEMPORARILY CRASHED, LOGIC SAVES, and MEMORY OVERLOAD. A man from Chicago wrote that he'd give an arm for a shirt that says PLEASE DON'T SQUEEZE THE SOFTWARE, and that he'd give an arm and a leg for one for his wife. My bank doesn't deal in that particular currency (except for the occasional pound of flesh). One fellow said he needed more shirts because his Bionic Toad was lonely ... I await each day's mail eagerly to see what interesting new input it will bring.

At least 50% of the ideas I receive are rather sexually suggestive such as: BYTE MY BAUD, FLOPPY DISCS, DISCREET COMPONENTS, and BAUDY BODY. I haven't chosen to print any of the raunchier suggestions, not out of prudishness (as those who know me well can confirm), but for purely (oops) practical reasons. They are the hardest to sell because many people are uncomfortable wearing them. For example, last year I wore my DATA INSUFFICIENT shirt to the NCC in New York. I didn't think it was suggestive, but after the 10th time someone said "Your data doesn't look insufficient to me, baby," I decided to discontinue printing it.

The shirts I have decided to print in time for Christmas ordering are: SWAPPED OUT on light green sent in by Dan Goodman of Somerville, NJ; DO NOT STAPLE SPINDLE FOLD OR MUTILATE on tan (naturally) suggested by Al Brunelle of Woonsocket, RI and also by Ron Walter of Chelsea, MI. UPWARD COMPATIBLE on red by Marcelle Gaig of New York City; TIME SHARING on navy by Kevin Frye of Old Orchard Beach, ME and also by William Amon of Harrisburg, PA. SPECIAL CHARACTER on gold by Robert Pallack of W. Palm Beach, FL, and for kids only MICRO BUG on yellow by Jane Foust of Marina, CA.

I have always enjoyed getting mail, so suggestions continue to be welcome. Anybody whose idea gets printed gets a free shirt with their phrase on it. In case of duplications the first two people with the same basic (oops again)) idea will receive free shirts, but I've got to stop somewhere.

Readers can find my shirts at a growing number of computer retailers, or may order directly from me, at $6 each plus .60 postage and be sure to include chest size.

Thanks again for the write-up and for helping me share the fun with your readers.

Martha Herman
114 West 17 St.
New York, NY 10011

Shorter and Faster ELIZA

Dear Editor:

Steve North's version of ELIZA programmed in MITS BASIC (July/August 1977) is up and running in our ALTAIR. It seems to be interesting to everyone who sees it. One important change though will allow ELIZA to run with nearly 2K bytes less memory: Delete line 90 which allocated array space for strings. These are never used since in MITS BASIC each non-dimensioned string variable and each member of a string array can each contain up to 255 characters.

For use on MITS version 4.0 EXTENDED BASIC, ELIZA will run faster if the following two changes are made:

1. DEFINT A-Z to define all non-string variables as integer and
2. Replacing lines 315-350 with a line incorporating INSTR the string search function
   330 T = INSTR(I$,K$):   IF T THEN S=K: F$=K$:
   GO TO 365.

Alan R. Miller
Dept. of Metallurgical & Materials Eng.
New Mexico Tech
Socorro, NM 87801

STAR TREK on a 360?

Dear Editor:

Since we were bitten by the Startrek "bug" at the Anaheim NCC (where we first subscribed to *Creative Computing*), it has been our goal to get Startrek to run on our computer, (360/30 with DOS/BTAM).

Well, would you believe Startrek in rpgII? Online?? In 20K??? (well, 4 programs of 20K)

Our problem is this: we would like very much to have a *BASIC* interpreter or compiler that we could run on our computer so that we could use *BASIC* programs direct, instead of rewriting into another language. We need a source listing that we can convert into a 360 language.

We all enjoy Creative Computing, but would like to see more of the complex simulation games.

Daniel H. Jones
Director of Information Services
Bardon Inc.
5405 Jandy Place
Los Angeles CA 90066

put...input/output...in

Inventor of the Computer

Dear Editor:
In your July-August issue, you reprinted excerpts from a May 1974 presentation by Dr. Carl Hammer (page 56). I believe you should correct an erroneous statement made at the opening of that paper.

In the second paragraph, Dr. Hammer gives credit to Drs. Eckert and Mauchly for the invention of the digital electronic computer. If I may quote the words of Federal District Court Judge Earl R. Larson in his decision: "Eckert and Mauchly did not themselves first invent the automatic electronic digital computer, but instead derived that subject matter from one Dr. John Vincent Atanasoff."

The story of the Atanasoff invention of the electronic digital computer and of his communication of that idea to Dr. Mauchly, can be found in the February 1974 issue of *Datamation*. The article in *Datamation* makes it clear, by quoting from Mauchly's correspondence, that indeed the idea was originated by Dr. Atanasoff. It is not a small point, and I believe you have a responsibility to set the record straight.

Michael W. Ham
Director, General Systems
The American College Testing Program
P.O. Box 168
Iowa City, IO 52240

Help for a PDP-8 System

[Ed. Note: Richard Brown's letter in the July-Aug issue asked for help on a PDP-8 system, including how to choose the right modems that would allow a Teletype to be moved around; how to hook up the modems; decide whether to interface floppy disk to the system; decide whether to interface inexpensive peripherals; and write handlers for peripherals. John Blake sent us a copy of his letter to Mr. Brown.]

Dear Mr. Brown;
Your letter to Creative Computing asks for a solution which does not exist. The problem does not compute.

1. Moving Teletypes? They don't like it and usually require a visit from the man from the Teletype Corp. to put them back into condition. You will need one at the computer to use as the console. Find some permanent places for the other three.

2. From this distance I can not tell if your "two-digit" telephone system is an overgrown intercom (with one side of the line grounded), or a small telephone system (with the lines balanced to ground. If the former, no modem available will work. If the latter, the modems will be about $250 each, and you need six. For this amount of money you can buy a microprocesser, and use it to solve the rest of your problems.

3. Both the Teletypes and the serial I/O devices are available with an RS 232 interface, or a 20-mA loop interface. The 232 interface has a maximum distance set by the cable capacitance: 25 feet for normal cable, and up to about 100 feet for special cable. The 20-mA loop interface will work over several miles of fence wire. You therefore use the 232 interface for the Teletype used as system console, and the 20-mA interface for the remote ones. There is probably enough spare wire in the existing telephone system to allow the hardwiring except for the run into the computer room. If you want to connect to "(or any phone system)" you are going to need a specialist, and a local one at that.

4. Interface the micro to the PDP-8 using a DEC-built serial, and the micro serial. This should be an RS-232 and should be able to run at 19,200 baud which will make it practically transparent for both computers.

5. I don't know if you want to use the DEC as the main computer, or the micro as the main computer. The advantage of using the core in the DEC for handlers is that it will still be there when the lights come back on.

6. If you buy peripherals for the S-100 bus on the micro, they will come with their handlers available.

7. You will find that the 20K of PDP-8 core gets used up by programs faster than Z-80 core in the micro — this suggests that you use the micro as the main computer.

John T. Blake, P.E.
P.O. Box 538
Yucca Valley, CA 92284

Interfacing an IBM 2311 to an 8080?

Dear Editor:
Has anyone interfaced an 8080 to an IBM 2311 disk drive? I have two of them but am personally software-oriented and can't find anyone locally to help me out. Any kind of assistance (even if it's to tell me it can't be done) would be appreciated.

Bob Stek
19 Mayfield Rd.
Regina, Saskatchewan
Canada S4V 0B7

Programming for the Handicapped

Dear Editor:
Enclosed you will find a check and mailing addresses for six subscriptions to *Creative Computing*. These subscriptions are graduation gifts to students of the first graduating class in computer programming for the physically handicapped at Lakeshore.

The course is the third to begin in the nation and the first to be associated with a university (of Alabama in Birmingham). It is a practical, vocationally-oriented course concentrating in COBOL programming with experience in IBM's TSO, JCL, Utilities, and the BASIC programming language.

We thought you might be interested in the existence of the course (and the reason for the subscriptions). We feel that computer programming is an especially appropriate occupation for the physically handicapped as well as an interesting avocation. We believe your magazine will help maintain their interest.

C. Joseph Williams
Lakeshore Rehabilitation Facility
3800 Ridgeway Drive
Birmingham, AL 35209

Anyone for Computer Bridge?

Dear Editor:
Has the new generation of home size computers reached the level where the game of Bridge can be programmed?

I am interested in a level of sophistication where a team of persons plays against another pair represented by the computer.

As in a "live" game, a period of bidding should precede the play of the hands.

And what about the possibility of one person playing with the computer as a partner while at the same time the computer represents an opposing team? Such a program would be particularly useful for teaching and for upgrading one's bridge skills.

Victor Kay
3650 Los Feliz Blvd.
Los Angeles, CA 90027

Pub. note: Victor wanted us to run a contest for the best bridge-playing programs. However, after our experience with contests two years ago, we're not anxious to run any more. Nevertheless, if you have a good bridge-playing program, please submit it to us for possible publication.

A Second Look at Calculators

Dear Editor,

Like E. R. Tufte in *Creative Computing* (May-June 1977, p. 34), I have 5 calculators in the house, 2 of which I consider lemons (though all of them work as advertised), and I have also encountered the feeling of guilt over my cumulative investment (which besides might have bought me a "luxurious" microcomputer system, for over $1270). While his observations on electronic obsolescence are a fair warning to uncreative computing fans or fellow travelers, most of your space could have been used more constructively, to suggest rather the following:

1. Printing is *not* worth it, because one can recall registers or repeat calculations as often as one wants, and one can program "split displays" of several numbers in one display window, or sequential displays of whole arrays, if desired with identifying subscripts flashed briefly before each output. What would you do with all the printout, if it is not labeled, and not on a page you can file or publish, anyway? (Buying a non-programmable with a printer built in is a particularly self-defeating combination, as standards in electronics change fast, and printer mechanics are stagnant and expensive.)

2. A good up-to-date card-programmable calculator is *not* likely to be "purchased with good intentions, and used once or twice," by anyone interested in creative computing. And I have seen players of ready-made games of skill return to Moon Landing or Battle Ships more than twice even on a less convenient key-programmable.

3. Calculators are *not* limited to engineers and business people, as one can see in the HP Users Library, which picked up about 5000 well-documented and distinct programs from owners of just one calculator model, in 3 years. Only about 8% were business, and about 34% engineering oriented. (Incidentally, as a note on software-sharing sociology, the world's most famous minicomputer, the PDP-8, scored less than 1000 contributed programs in the DECUS Library, in 6 years.)

The problem of "compulsive investor's guilt" was solved for me in 2 ways. First, since 1974 my family and I have written up 21 programs we thought noteworthy, and got them all accepted in the internationally accessible HP-65 Users Library, and we have quite a few more in the (now congested) pipeline to the HP-67 Users Library. Secondly my wife, a biologist, did all computing for publication of her master's thesis, and for preparation of her custom-written programs, and enjoying absolute convenience and flexibility of access at her desk, near-perfect efficiency, and not the slightest distraction from hardware reliability problems.

Paradoxically, I don't continue to use any of the "big" programs that were so exciting to write and shake down. Professor Tufte's comparison with Dr. Samuel Johnson's dog walking on its hind legs seems stunningly perceptive in this connection. However, is it really unnatural to use a handy toy to model computational processes on a small scale, for more intimate familiarization? Or to break the tedium of reading a mathematical or technical textbook by breezing through examples which the editor may have found too laborious to check for accuracy? And is it not a stimulating mirror for one's mathematical fitness and quantitative applications ingenuity, independent of academic conventions, if one has occasion to see what turns up, besides games, in a calculator users library? In addition to the more spectacular competitive computing sportsmanship, which is typically elicited by programmable calculators, the keen observer will get a moderate traffic of little cases in daily life, which we still have to learn to recognize as instances of programmable computation; e.g., converting data to percentages which manifestly add up to exactly 100.

Although I got burned twice already, in a way, I frankly look forward to discrete little machines with more capabilities, to get hands-on experience with more challenging mathematical procedures like prediction filtering (applicable in Battle Ships like in Norbert Wiener's anti-aircraft guns), pattern recognition, feature selection, simulation of multivariate random processes, and ultimately, management of my appointments and agenda diary, with the whole show remaining easily portable like a tricorder in STARTREK. Some programs recently completed in shirt-pocket format are Nonlinear Regression, Factor Analysis, Multivariate Analysis of Variance, Exam Grading with Item Analysis, Questionnaire Tabulation, Dynamic Programming, and Master Mind, although not all of these exhibit dazzling performance and range with current equipment.

I will barely take the advice of waiting till the new models appear in the discount stores, and I think the manufacturers are quite justified to plan for "More Power in Your Shirt Pocket or Skirt Pocket."

R. Belling
398 Ogden Ave.
Jersey City, NJ 07307

Sloppy Disks

Dear Editor:

Our company is a manufacturing and sales outlet for a new generation of micro and mini computers. We have a franchise aim for a hobby computer store with a strong emphasis on the small business market.

During "Ma Bell's" yellow-page campaign, our company produced two ads showing the internal structure of a business retail and a hobby retail division. As these two ads were given the same address for billing, the printers combined the ads and came up with the enclosed yellow page ad. As you can see, our company is now the proud (?) producers of the only *SLOPPY DISKS* in America for the next year that are double density formated.

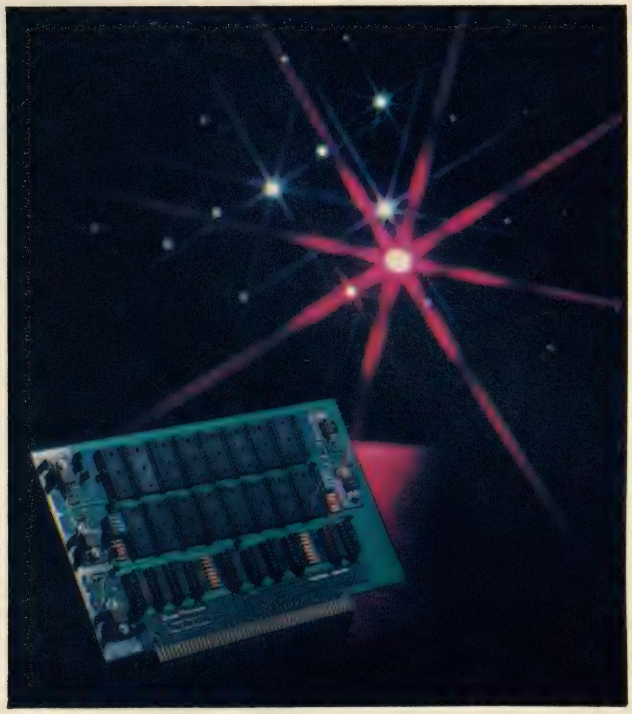
We figured that such an engineering feat as *SLOPPY DISKS* would be of interest to your public. Anyone interested in our design structure will be sent an information package.

Douglass C. Boseman
President
ComputerCo, Inc.
5849 Dorchester Rd.
Charleston Hts., SC 29405

PROBLEM SOLVER SYSTEMS, INC.

Manufacturers of QuanTronics Computer Products

Space Age Memory...



...at Down to Earth Prices

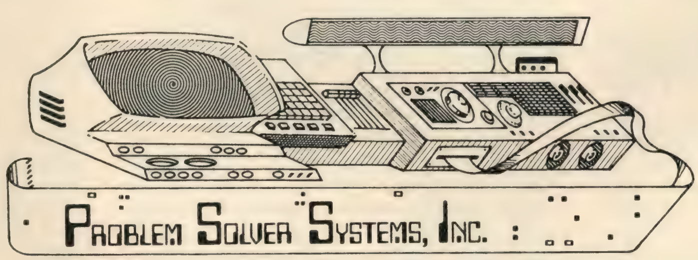
MM8 Static RAM

- Available in 250nS and 450nS versions
- S-100 Bus Computer Systems
- Memory capacity of 8K-bytes x 8 bits maximum
- Addressing selectable by DIP Switch
- Wait cycles selectable by DIP Switch
- Memory protect from ¼ K to 8K by DIP Switch
- Data output, address input lines fully buffered
- Provision for SOL "Phantom" line

MM16 EPROM

- Utilizing up to 16 2708 EPROMS
- S-100 Bus Computer Systems
- Memory capacity of 8K or 16K bytes by DIP Switch
- 8K boundary addressing by DIP Switch
- 0 to 4 wait cycles by DIP Switch
- Data output, address input lines fully buffered
- Hi-grade glass-epoxy with plated-thru holes
- Epoxy solder masked

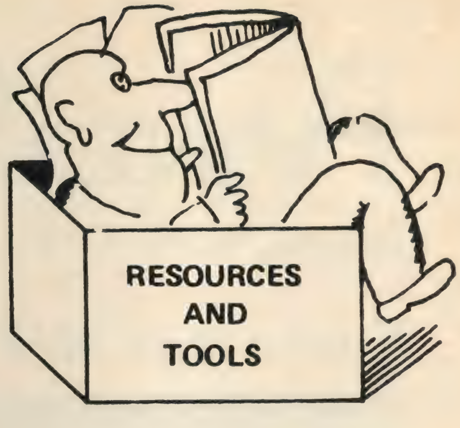
Find Out For Yourself — See Your Local Dealer Today!



Manufacturers of QuanTronics Computer Products

8040 DEERING AVE. SUITE 10 CANOGA PARK, CA. 91304

COMPLEAT COMPUTER CATALOGUE



We welcome entries from readers for the "Compleat Computer Catalogue" on any item related, even distantly, to computers. Please include the name of the item, a brief evaluative description, price, and complete source data. If it is an item you obtained over one year ago, please check with the source to make sure it is still available at the quoted price.

Send contributions to "The Compleat Computer Catalogue," *Creative Computing*, P.O. Box 789-M, Morristown, NJ 07960.

BOOKS AND BOOKLETS

COMPUTER BASIC SELF-INSTRUCTION COURSE

Heath Company has introduced its EC-1100 self-instruction course in BASIC language programming techniques. The course, according to a company spokesman, takes an approach which has been designed to teach even those with little or no computer experience the skills necessary to intelligently converse, create and program in the BASIC computer language.

Programmed-instruction texts combined with practical demonstration programs and practice problems are employed in the EC-1100 course to teach BASIC language formats, commands, statements and procedures. In addition, the course goes one step beyond conventional courses and texts, and gives the student a practical introduction to the creative and problem-solving aspects of programming in BASIC.

The EC-1100 is keyed to the Heath computer systems; however, because of the universal machine-independent nature of the BASIC language, this course can be an ideal supplement to any other computer systems using BASIC. Mail-order, $29.95.

Heath Company, Dept. 350-450, Benton Harbor, MI 49022.

THE FIRST BOOK OF KIM

This collection of dozens of programs—some useful, some recreational, all tested and documented — runs on the basic Commodore KIM-1 system. Also included is a Beginner's Guide and other information useful to current and prospective KIM owners. $9.

ORB, P.O. Box 311, Argonne, IL 60439.

BIBLIOGRAPHY ON COMPUTER IMPACT

An annotated bibliography on Computer Impact on Society is available on microfiche from the Virginia Institute of Marine Science. The bibliography contains more than 2,000 entries of books, articles and other items. Most items are annotated. The listing is alphabetical by author; however, the entire system is stored in a hierarchical storage and retrieval system at the University of Wisconsin and special-purpose subsets of the collection may be obtained. The complete set is $2.30 for individuals in the United States, and $3.30 for others.

Gerald L. Engel, Dept. of Computing and Statistics, Virginia Institute of Marine Science, Gloucester Point, VA 23062.

DESIGNING WITH MICROPROCESSORS

This tutorial deals with the principles and practises of microcomputer design, covering such topics as chip architecture, software aids, development systems, microprocessor applications, networks, busing strategies, and distributed intelligence. To IEEE members, $7.50; nomembers, $10.

IEEE Computer Society, 5855 Naples Plaza, Suite 301, Long Beach, CA 90803.

MAGAZINES, JOURNALS

COMPUTER CHESS NEWSLETTER

The first issue of the Computer Chess Newsletter is an 8-page offset publication that includes information about chess programs for microcomputers, a letter from Bobby Fischer commenting on the Chess Challenger computer, an article on a "move tree," and a list of computer-chess books. Editor Doug Penrod says the next issue will be 16 pages long. Price is 75¢.

Computer Chess Newsletter, 1445 La Cima Rd., Santa Barbara, CA 93101.

VENDOR LITERATURE

DASHER TERMINALS BROCHURE

An 8-page brochure describing the Dasher printer and display terminals is available from Data General Corporation, Communications Services Department. The brochure describes the 60 and 30-cps keyboard send/receive and receive-only DASHER printers with numeric keypads, 128-character upper and lower-case ASCII character sets, and adjustable paper-feed tractors. The brochure also describes the Dasher displays featuring detached keyboards, 1920-character displays that tilt and swivel for operator convenience, and 11 user-defined function keys for high transaction throughput. Dasher printers and displays come standard with EIA RS232-C or 20-mA current-loop interfaces for compatibility with all Nova and Eclipse computers. Dasher terminals are also Teletype compatible for local or remote communications with other computers and operating systems.

Data General, Route 9, Westboro, MA 01581.

MICROCOMPUTERS & COMPONENTS

Cybertronics is one of a small number of companies offering both component parts for the design-your-own computer, and standard computer kits. In components, Cybertonics sells ICs from AND gates to microprocessor chips, plus crystals, diodes, capacitors, switches, wire, etc. Design and troubleshooting aids include breadboards and probes. Mainframes and peripherals include the IMSAI 8080, iCOM floppy disk, CP 10 video terminal, Centronics printers, etc. A 480-page catalog describes these and many more products in detail, for $1.50.

Cybertronics Inc., 312 Production Ct., Louisville, KY 40299. (502) 499-1551.

CATALOG OF BREADBOARDING AND TEST EQUIPMENT

Continental Specialties' latest hobbyist catalog offers breadboard and test equipment for the hobbyist and the professional. Includes are breadboard sockets, pre-assembled breadboards, logic monitors and probes, function and pulse generators, R/C bridge, test clips, and components.

Continental Specialties Corp., 44 Kendall St., P.O. Box 1942, New Haven, CT 06509. (203) 624-3103.

E & L CATALOG

A 24-page catalog of circuit design and educational aids from breadboards and textbooks to a microcomputer development station is now available from E & L Instruments. The "Complete Bugworks" catalog illustrates and describes more than 155 products for the designer, researcher, hobbyist and student. Specifications, prices and ordering information are included.

The contents include solderless sockets, breadboards and accessories for "plug-in" wiring as well as the Innovator series of plug-in functional modules called "outboards." These modules speed up breadboarding and experimental setups and make learning easier and more fun. The selection of 56 outboards, assembled or in kit form, includes the most frequently used circuit functions. Designer "breadboxes" combine the advantages of solderless wiring with the convenience of self-contained power supplies, controls and displays. Some are for home and classroom study use; others aid in the specialized breadboarding of opamp, digital-logic and microprocessor circuitry. Designers are also available assembled as well as in kit form.

E & L hardware is ably supported by a select list of coordinated educational texts, manuals and reference books. Subjects range from elementary electronics to computer design and programming. Among the 16 texts are the latest editions of the famous Bugbooks, combined texts and lab manuals.

New from E & L instruments this year are a low-cost PROM programmer and two hexadecimal keypads for use with the popular MMD-1 Mini-Micro Designer, the 8080A-based microprocessor training and hardware development system.

E & L Instruments, Inc., 61 First St., Derby, CT 06418. (203) 735-8774.

COMPUTERS

POLY 88 SYSTEM SIXTEEN

A ready-to-run system, the POLY 88 System Sixteen lets you solve those home financing problems, perform a statistical analysis, or enjoy a host of challenging games. The 16K system features a high-speed video display and an alphanumeric keyboard. Cassette tapes are used for permanent program storage. Programming is made simply by the BASIC software package. PLOT and TIME are two of the unique features which rely on our video graphics and real-time-clock. Other features include VERIFY so that you know that your tape is good before you load another. Scientific functions, formatting options, and string capabilities are also included. In addition to the programs written by the user, the POLY 88 program library makes a growing number of applications available to the POLY 88 owner. Assembled, $2250; Kits start at $735.

PolyMorphic Systems, 460 Ward Drive, Santa Barbara, CA 93111. (805) 967-2361.

AJ SMALL-BIZ SYSTEM

Anderson Jacobson, Inc. (AJ) a data communications company, introduces its latest product, the AJ 1500 Small Business System, featuring multiprogramming capability that can accommodate up to eight terminals interactively. Included in the basic AJ 1500 system is 32K bytes of core memory, two 630K double-density diskette drives and a 45-cps daisy-wheel I/O console printer. Optionally, the AJ 1500 system may be expanded to include 64 Kbytes of core memory, four diskette drives, and four 10-megebyte cartridge disk drives. AJ also offers an optional choice of console printers; select either a 60 or 120-cps dot matrix I/O printer or a full-screen CRT. A 300-lpm printer is also available.

AJ 1500 system offers standard client accounting software designed for CPA and public accountants. Designated CPA III, software programs include general ledger, budget accounting, payroll (non-check writing), and loan amortization. To these may be added optional packages such as accounts receivable, accounts payable, and payroll (check writing) as well as a text editor for word processing. About $22,500 including CPA III software.

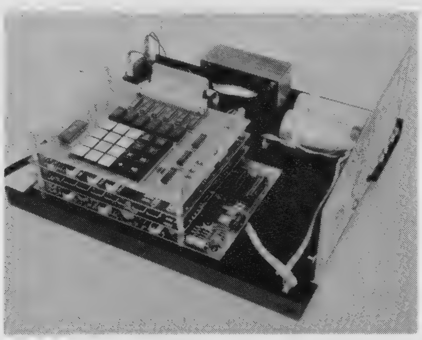
Robert Reed, Computer Systems Div., Anderson Jacobson, Inc., 521 Charcot Ave., San Jose, CA. (408) 263-8520, Ext. 372.

LOW-COST DEC COMPUTER FOR STUDENT USE

Digital Equipment Corp. announced a low-cost instructional computer system designed to extend computer access to large numbers of students at the secondary level. The MSB-11 (Mark Sense Batch-11) is a compact batch-processing system that supports both BASIC and FORTRAN IV programming languages and uses student-prepared mark sense cards as its primary input medium.

The PDP-11/04-based system, with 32K bytes of semiconductor memory and a dual floppy-disk unit for program and data storage, is housed in a short (4 ft. high) cabinet. Input/output devices include a 180-cps line printer, LA 36 DECwriter II console terminal, and a new mark-sense card reader, the CMS-11K. With a subset of the RT-11 operating system and the BASIC language, the MSB-11 system is priced at $21,460.

Digital Equipment Corp., Maynard, MA 01754.



Z80-BASED MICROCOMPUTER

Martin Research, has a Z80-based system, the MIKE 8, the newest entry in the firm's Modular Micro series of compact (5½-by-7-inch) microcomputer boards. The Model 882 system comes with 4K of RAM, a 1K monitor program in a PROM, the CPU board, and a "console board" which has a calculator-type keyboard and six LED digits. The monitor allows the user to enter and execute programs via the console, and offers advanced debug features, including RAM test, single-stepping, and setting traps.

The system includes a PROM programmer, so the user can permanently store his programs in a blank 2708 PROM (which is included). An ultra-violet lamp, for erasing PROMs, is supplied. The Model 882 is mounted on a base with its own switching-regulated power supply. A "black-box" enclosure allows the unit to be adapted to many industrial and hobby applications. Assembled, $895.

Martin Research, 3336 Commercial Ave., Northbrook, IL 60062. (312) 498-5060.

LOWER ALTAIR PRICES

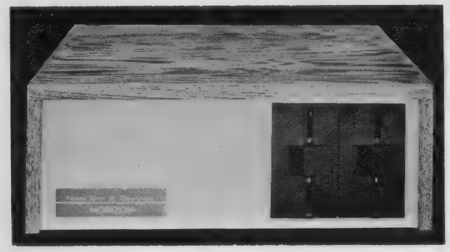
Manufacturing efficiencies now being carried out by Pertec Computer Corp., the new owner of MITS, are expected to cause such savings that, in anticipation, Pertec is "planning to pass on price reductions of about 20 percent on a variety of components in a personal computer system." The 8800B kit, which was $875, is now $750; the 8800B assembled with 18 board slots, is down from $1250 to $1070. And similar price reductions are in effect for turnkey versions of the 8800B, and for the 680B. The stated programs that will be installed to "immediately improve the cost-to-retail-sales-price-relationship" include "better financial controls, new manufacturing techniques, improved material handling and inventory processes."

MITS, Inc., 2450 Alamo S.E., Albuquerque, NM 87106. (505) 243-7821.

WORD-PROCESSING SYSTEM

Computer Power & Light, Inc. announces what is said to be the first commercial-quality, microcomputer-based word-processing system for under $6,000, complete. Based on the field-proven COMPAL-80 computer and Xerox Corp.'s Diablo 1620 daisywheel printer, it contains features "found only on systems costing $20,000 or more." Among these are: complete text editing on a large CRT; insertion or deletion of text, and the ability to move blocks of text anywhere; variable-speed scrolling of entire text on the CRT, forwards and backwards; ability to search for all occurrences of a specific word or group of words and replacement with alternative word or words; storage and retrieval of finished text on low-cost Phillips audio cassettes at the rate of 240 cps; a variety of printing options, including variable line length, 1-5 spaces between lines, variable character spacing, presettable page headings, page numbering, and right and left-margin justification using the Diablo's character-spacing routines — no extra blanks are inserted in your text, nor is there any need for hyphenation.

Computer Power & Light, 12321 Ventura Blvd., Studio City, CA 91604. (213) 760-0405.



HORIZON COMPUTER WITH INTEGRATED DISK

The new North Star Horizon computer uses a full-speed (4-Mhz) Z-80 microprocessor and includes 16K bytes of memory, a disk controller with one or two Shugart minifloppy disk drives, and full extended disk BASIC. A serial I/0 port is included for connection to any standard baud-rate terminal. Options for the Horizon computer include additional disk drives, hardware floating-point arithmetic board, 24-line by 80-character upper and lower case video display controller (VDC) board, and 16K memory board with parity check. The Horizon computer uses the S-100 bus.

The single-drive Horizon-1 sells for $1599 in kit form and $1899 assembled. The dual-drive Horizon-2 sells for $1999 in kit form and $2349 assembled.

North Star Computers, Inc., 2465 Fourth St., Berkeley, CA 94710. (415) 549-0858.

ORGANIZATIONS

NASAGA

The basic goal of the North American Simulation and Gaming Association (NASAGA) is to advance an optimal, responsible application of the technique of simulation and gaming. The objectives of the Association are: to facilitate communications among persons interested in the field of simulation and gaming; to promote the training of specialists in the field of simulation and gaming; to facilitate communication between these specialists and policy-makers, students, and other concerned persons; and to promote the development of better techniques in the field of simulation and gaming.

Annual membership in NASAGA begins July 1 each year. Anyone becoming a member for 1977-78 at a cost of $12 will receive: a membership directory; a special NASAGA members rate at the 1977 NASAGA conference; a membership card; a one-year subscription to *Simulation/Gaming;* and an opportunity to contribute to expanding the use of gaming and simulation.

NASAGA % COMEX — Davidson Conference Center, University of Southern California — University Park, Los Angeles, CA 90007. (213) 741-6569.

TERMINALS

SPACE-SAVING TERMINAL

A small alphanumeric display terminal designed as an alternative to a CRT is now available from Computerwise, Inc. The Transactor I Data Terminal consists of a single-line, 32-character gas-discharge display with a 5 x 7 dot matrix, and a 53-key Teletype-style keyboard. It can be attached to almost any computer, with an RS-232 or 20-mA current-loop interface. Switches allow the user to select the operating mode including; 110-9600 baud rate, full or half duplex, even/odd/no parity, five to eight data bits, and one or two stop bits.

Lightweight and small, the Transactor is housed in an aluminum case that measures 6 in. high x 15 in. wide x 11 in. deep. A stylized molded case is available. The standard Transactor I in quantity is $595.00.

Computerwise, Inc., 4006 East 137th Terrace, Grandview, MO 64030. (816) 765-3330.

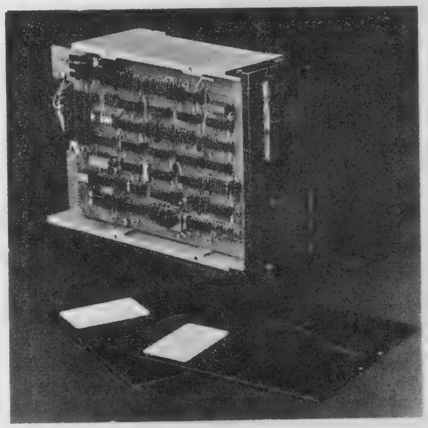
PERIPHERALS

DUAL MINIFLOPPY KIT UNDER $1,000

The first dual minifloppy kit said to be under $1,000 has been announced by Southwest Technical Products Corp. The $995 MF-68 minifloppy disk system is designed for use with the SWTPC 6800 computer system. It includes all hardware and software needed to provide a complete high-quality dual-disk operating system. The system can be expanded to its four-drive limit with an $850 MF-6X expansion kit.

The extensive software provided with the MF-68 includes both disk BASIC and a floppy-disk operating system (FDOS). Operating-system commands make disk operation, including copying all or any part of one diskette onto another, simple and easy. Commands include CREATE, SAVE, RUN, LOAD, PURGE (delete), PACK, CATALOG, RENAME, INITIALIZE and PATCH. The MF-68 is offered in kit form only. It includes controller, chassis, cover, power supply, interconnecting cables, assembly instructions, two assembled Shugart minifloppy drives, and a diskette with the FDOS operating system and disk BASIC. The MF-6X expansion kit includes power supply, chassis, cover, and two assembled minifloppy drives. Prices of $995 for the MF-68 and $850 for the MF-6X includes prepaid postage.

Southwest Technical Products Corp., 219 W. Rhapsody, San Antonio, TX 78216.



DOUBLE-DENSITY MICROFLOPPY DISK DRIVE

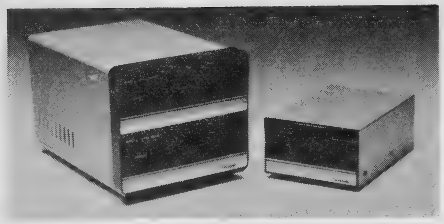
The first flexible disk drive for 5-1/4″ diskettes to offer double-density recording of 250,000 bytes on each side of a diskette, was introduced this year.

The drive, designated the Pertec Model FD200 Microfloppy, is mechanically and functionally interchangeable with the diskette drive introduced earlier by Shugart Associates but permits writing on 40 tracks versus a 35-track limit for the Shugart SA400 unit, in addition to the double-density feature. The Pertec FD200 also allows recording on both sides of a diskette, whereas other units are limited to one side, according to Pertec..

Both the signal interface connector and the DC power connector are compatible with the Shugart equipment, and mounting holes and outline dimensions are the same. Up to four FD200 drives can be "daisy-chained" on a single 34-line ribbon cable. The new Microfloppy drive also employs an IBM-compatible read/write head with "tunnel erase." Head life is estimated at 20,000 hours. A special design feature prevents the head from disengaging from the positioning cam mechanism. The FD200 incorporates a molded plastic faceplate and door. Light-emitting diodes on the faceplate indicate the drive that has been selected.

Formatted capacity is 102,400 bytes per side. The data transfer rate of the diskette drive is 125,000 bits per second; average access time is 463 milliseconds. Recording density at the inside track is 2581 bits-per-inch. $405.

PCC/Pertec, 21111 Erwin Street, Woodland Hills, CA 91367. (213) 999-2020.



FLOPPY FAMILY WITH 5¼-INCH FORMAT

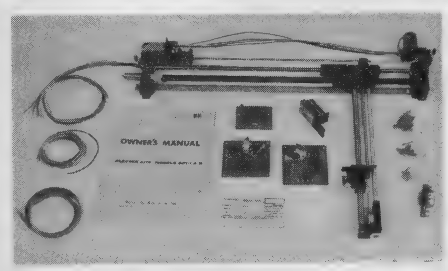
A family of fully integrated floppy-disk systems from Micropolis Corporation is claimed to be the first to package the performance and storage capacity of 8-inch disks in a 5¼-inch format, at the same price as comparably-sized units. Designed to make the 5¼-inch format viable in cost and performance for OEM's and home computer hobbyists alike, the MetaFloppy family of four systems offers the additional advantages of plug-in microprocessor compatibility and a complete BASIC software package. They are available in single and dual drives with capacities ranging from 143 to 630 kilobytes.

The smallest MetaFloppy system, the 1043-Mod I, is a single-drive using a 35-track disk with a capacity of 143 kilobytes — as compared to 70K bytes for other 5¼-inch units — at $945 in quantity one, which includes power supply, controller, interface cable, and BASIC software.

The Model 1043-Mod II system uses a 77-track disk with a 315-kilobyte capacity and a price of $1095, including power supply, controller, interface cable, and BASIC software.

Two compact dual units round out the family. The Model 1053-Mod I stores 286 kilobytes at $1545 while the Model 1053-Mod II packs 630K bytes and a $1795 price. Both include power supply, controller, cable, extended disk BASIC and power supply.

Micropolis Corp., 9017 Reseda Blvd., Northridge, CA 91324.



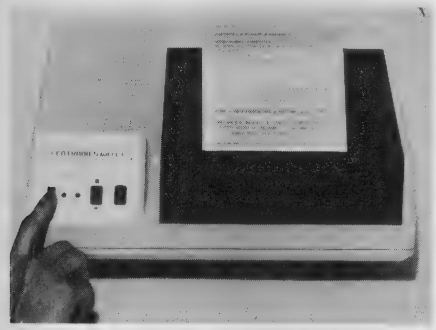
LOW-COST X-Y PLOTTERS

Sylvanhills Lab announces 8080-based software to control their series of plotters. This enables the microcomputer to act as the controller for the plotter and requires about 2K of memory. The software format is such that it may be used in conjunction with application routines available from Micro-Visions Inc., 4926 Travis, Houston, TX 77002. Plotters are shipped completely assembled and tested, but require the purchaser to mount them on his drawing surface and interconnect between the control PC boards and his computer. Requires an 8-bit parallel I/O port and 5 and 24-volt power sources.

Applications include architectural,

mechanical, and schematic drawing; PC board artwork; positioning of small objects; computer generated art; games; and many others. Sizes available are 11 x 17 ($750), 17 x 22 ($895) and 22 x 34 ($1200).

Sylvanhills Lab. Inc., #1 Sylvanway, Box 239, Strafford, MO 65757, (417) 736-2664.



HIGH-SPEED MICROPRINTER

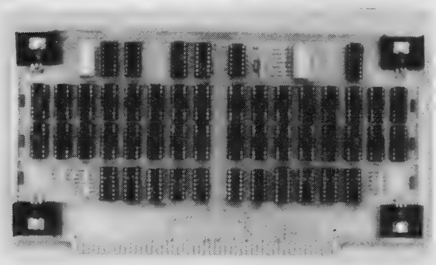
Centronics Data Computer Corp. has a high-speed, low-cost, compact microprinter. Aimed at the home, hobby, and microprocessor markets, the 240-cps Micro-1 is offered as a complete unit including case, power supply, 96-character ASCII generator and interface, paper-roll holder, low-paper detector, bell, and multiline asynchronous input buffer.

The microprinter produces copy on aluminum coated paper by discharging an electric arc to penetrate the coating, which is less than one micron thick. Toners and ribbons are not required. The printed characters, unlike those resulting from thermal printing, are impervious to light, temperature, and humidity. In addition, the finished printed page may be reproduced on any office copying machine.

The microprinter electronics allows the machine to produce copy at a rate of 180 lines per minute on 4 ¾ inch roll paper and provides the user software selection of 20, 40, or 80 columns. $595.

Centronics Data Computer Corp., Hudson, NH 03051. (603) 883-0111.

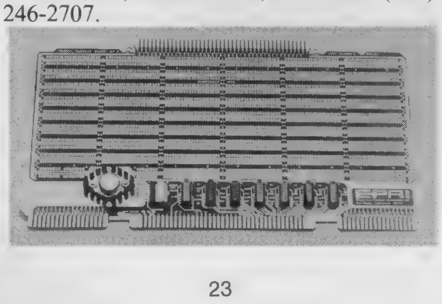
MISC. HARDWARE



CYBERCOM 4K STATIC PROM/RAM BOARD

The new MB9 4K static PROM/RAM board, from the Cybercom division of Solid State Music, features addressing at any 4K block segment with fully buffered address and data lines. The jump-on-reset circuitry allows execution at any 1K boundary or may be modified to jump on power-up. DIP-switches located at the top of the board are for handling jump address and enable, board address, memory protect, and 0, 1, or 2 wait states. RAMs and PROMs may be mixed in any 256-byte increments. $79.95 without memory.

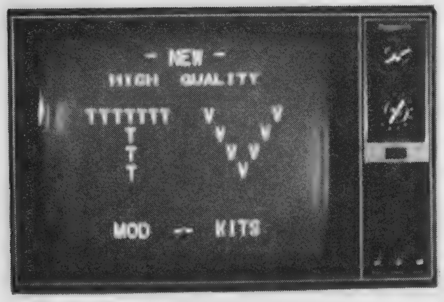
Cybercom/Solid State Music, 2102A Walsh Ave., Santa Clara, CA 95050. (408) 246-2707.



3000-HOLE PROTOTYPING BOARD

Electronic Product Associates announces a new general-purpose prototyping board for use in the Micro-68 microprocessor systems. The 8″ x 14.8″ (20.32 cm x 37.59 cm) GP-2 board is exorcisor-bus compatible and has complete bus buffering already established onboard with 8833 driver/receivers. The GP-2 board contains Vcc and ground busing, 3000 holes worth of blank DIP patterns for thirty-five 24, 40 or 42-pin DIP packages, or 107 14 or 16-pin DIP packages. $170.

Patti Neumann, Vice President, Electronic Product Associates, Inc., 1157 Vega St., San Diego, CA 92110.



VIDEO MODULATORS

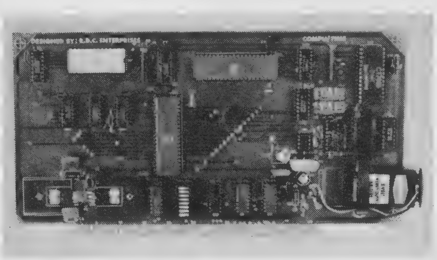
Convert a commercial TV into a monitor. Normal TV reception, when desired, is un-affected. Vamp, Inc. offers the computer hobbyist or video enthusiast a choice of three kits. Each kit comes with complete assembly and installation instructions. All of the kits will work with either Black & White or Color television sets, and all are perfectly safe when installed as directed.

TRVM ($20.95) kit is for transformer-isolated sets only. A buffered non-inverted video signal (some transformered sets require inverted video, therefore, you should use HCVM) is fed to your set's video amp, thus bypassing the tuner and I F Sections. Very high resolution is possible in this manner.

HCVM ($24.95) is a universal kit. It will work on any set (especially "hot chassis" types). It provides both inverted and non-inverted video, and a good isolation between your video source and your TV set's power supply.

RFVM ($9.95) is a video-only modulator which is operated on +5 Vdc at 1 mA and over a frequency range which allows you to choose channels 2 through 6. The RF output of the modulator connects to your set's antenna terminals.

Vamp, Inc., P.O. Box 29315, Los Angeles, CA 90029.



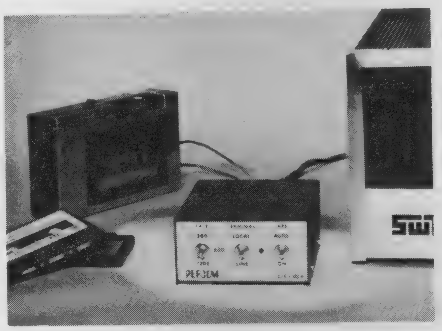
CLOCK/CALENDAR AND SCIENTIFIC CALCULATOR ON ONE PC BOARD

This S-100 bus compatible PC board marketed by COMPU/TIME of Huntington Beach, CA provides two independently separate functions one a single PC board. The Digital Clock/Calendar function has a crystal-controlled time base for providing date or time output in real time for purposes such as time and date stamping of output listings, memory dumps, or CRT Display. Two setable coincidence counters are incorporated to provide elapsed time capabilities for alarm or timing applications. Time, date, and counters are set or changed by software. Once set, the Clock/Calendar does not require re-initialization. If power is shut down, a battery backup system is provided.

The 40-Function Scientific Calculator provides the microprocessor with a hardware solution to floating-point, trigonometric and algebraic problems as well as the basic math functions. Savings in memory can be realized and made available for other tasks while the hardware performs intricate computations or general mathematics via the calculator array.

Compu/Time is available in three configurations: (1). both time/date and calculator capabilities, model CT100, kit $199; (2) time/date only (coincidence counters are included). Model T102, kit $165; (3) calculator capability only. Model C101, kit $149.

Compu/Time, P.O. Box 417, Huntington Beach, CA 92648. (714) 638-2094.



CASSETTE/TERMINAL INTERFACE FOR SW 6800

PerCom Data Company announced what is said to be the first high-speed, self-clocking cassette/terminal interface for the Southwest Technical Products 6800 microcomputer.

Designated the CIS-30+, the dual-function unit interfaces cassette data at a user-selectable 30, 60, or 120 bytes per second, and provides RS-232 interfacing at 300, 600, or 1200 baud.

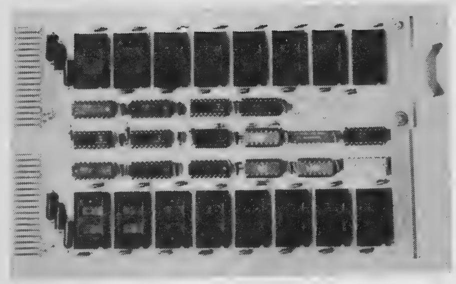
Cassette data is KC-standard/biphase encoded for dependable, self-clocking data exchange. The self-clocking feature virtually eliminates tape-speed variation errors, and permits the use of inexpensive audio recorders without loss of data reliability.

The CIS-30+ users manual and optional test cassette include PerCom cassette operating software; however, the 6800-resident Mikbug is the only software needed except for loading at 120 bytes per second. The CIS-30+ also plays unmodified SWTPC cassette software.

Cassette record and playback circuits are independent permitting operations such as cross-filing, and optional program control of recorders is available. This is accomplished without the use of a 6800 CT-CA cursor control board.

The CIS-30+ data terminal interface is full duplex, and provides for data exchange between the 6800 and the user's terminal or TVT at 300, 600, or 1200 baud. Kit, $69.95; assembled, $89.95.

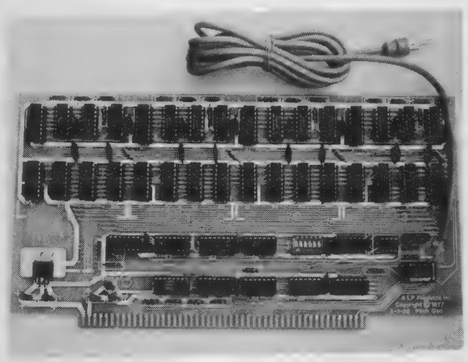
PerCom Data Company, 4021 Windsor, Garland, TX 75042 (214) 276-1968.



16-BIT EPROM MEMORY BOARD

The RMRV-8K is an 8K word by 16-bit EPROM memory board designed for use with DEC's LSI-11 microcomputer. It offers the highest density EPROM memory currently available for this popular microcomputer. The memory board occupies one dual-height module slot in the LSI-11 backplane. Packaging density is achieved by using the easily programmed and UV-erasable 2708, 8192-bit (1K by 8) memory IC. Addressing is jumper-selectable for any two 4K banks in the 0 - 28K address space. Bus handshake logic is handled in 1K segments allowing for 1 to all 8K to be enabled in reply to a memory-send request. $285, without EPROMS.

RDA, Inc., 5012 Place, Beltsville, MD 20705. (301) 937-2215.



MUSIC BOARDS

The 10-5-9 and 10-5-10 Quad Chromatic Pitch-Generator boards are designed as a low-cost start in computer-controlled music generation. The single-board Pitch Generator produces one to four tones simultaneously; two boards can be used to produce eight simultaneous tones in stereo. Each of the four tones are separately controlled and can produce any of 96 tones which form an 8-octave range. This covers the entire standard piano range, plus 8 higher pitches. Special connections allow later expansion with accessory boards to control various sound parameters. Using the optional on-board crystal oscillator or a 2-MHz source (external or pin 49 on the S-100 bus) all pitches are within 0.1% of the A=440 Hz standard. The 10-5-9 is S-100 compatible, and the 10-5-10 is compatible with parallel output ports. Kit prices for both versions range from $111 to $159 (depending on the number of simultaneous tones), the assembled price is $185. Oscillator is an additional $16. Available for product evaluation: data sheet (free), demonstration record ($1), and owner's manual ($3 plus $1 postage).

ALF Products Inc.; 128 South Taft; Denver, CO 80228. (303) 234-0871.

SOFTWARE

MATHPACK

Through the interactive use of a time-sharing computer system, Mathpack is a new approach to the old problem of teaching the basics of obtaining the correct answers to arithmetic and algebra problems. Mathpack consists of 121 sets of three computer programs, written in BASIC. Each set provides: (1) drill and practice, providing random sample problems, and no two students receive the same set; (2) program for checking answers, with correct answers provided; (3) testing.

Piedmont Call-A-Computer, Inc., Education Dept., P.O. Box 10234, Raleigh, NC 27605.

HIGH-LEVEL LANGUAGE FOR SC/MP

An easy-to-use microcomputer language, similar to BASIC, has been developed by National Semiconductor Corp. for its SC/MP microprocessor system. Called NIBL (pronounced "nibble") for National Industrial Basic Language, it has been placed, without charge, in the public domain to insure wide dissemination. It is available in paper-tape form through COMPUTE, the user-group newsletter.

NIBL is an adaptation of Tiny BASIC, and requires a minimum of 4K bytes of ROM and 2K bytes of RAM user space (allows about 60 average NIBL statements). As an option, NIBL will presently support an additional 26K bytes of user memory. Two versions of NIBL are available, for the earlier model and for SC/MP II.

Phil Roybal, National Semiconductor Corp., 2900 Semiconductor Dr., Santa Clara, CA 95051. (408) 737-5743.

PRE-RECORDED SOFTWARE

Compu-Quote announces its Computer Cassettes, a line of pre-recorded software available on high-quality, low-noise Philips audio cassettes. Four different cassettes are now available: 4K BASIC ($10), 8K BASIC ($15), VIDEO CHECKERS ($10, plays under MITS 8K BASIC, requires PolyMorphic board and 16K memory). and GAMES ($15, plays under MITS 8K BASIC, eight games including SEA WAR, WUMPUS, BLACKJACK, BAGELS, etc.).

All are recorded in Tarbell format, for 8080 systems.

Compu-Quote, 6914 Bergquist Ave., Canoga Park, CA 91307. (213) 348-3662).

EMPL/8080 INTERPRETER

EMPL, a micro APL for the Z-80/8080, is now available. The interpreter itself resides in 5.5K bytes, but a minimum of 8K is recommended. EMPL has numeric and character vectors, user-defined monadic and dyadic functions, 22 primitive functions, 9 system commands, and many other special operators and characters. EMPL can be run either in the ASCII or APL character set. The range is ± 32767, and double-byte integer arithmetic is used. EMPL is $10, including a Tarbell cassette and User's Manual.

Erik Mueller, Britton house, Roosevelt, NJ 08555. (609) 448-2605.

APPLICATION SOFTWARE BY MAIL ORDER

Custom microprogramming, BASIC and FORTRAN programs. Lists of ready programs, in a variety of application areas:

Compumax Associates, Suite 310, 505 Hamilton Ave., Palo Alto, CA 94301. (415) 321-2881.

MISCELLANEOUS

PERSONAL COMPUTER NETWORK

The PCNET (Personal Computer NETwork) Committee has been functioning in the Palo Alto area since the April Computer Faire. The committee's goal is the creation of regional (followed by national) personal computer networks for the computer-to-computer transfer of messages and files. A set of network protocols (sets of conventions defining all

levels of intercomputer communication) is almost completely designed. These protocols should be operable in 8K bytes of machine code, and are designed to be implemented in string BASIC.

The committee believes this should be attractive to personal computer users. Participation will be voluntary; you can decide to participate (or not) on any given day of network operation. Network functioning will be relatively insensitive to the absence of an appreciable fraction of member computers.

Current thinking indicates the following tentative equipment required for participation in the network: A personal computer with 12-16K of RAM and string BASIC; an originate/answer modem capable of 300 bps.

For further information write or call: Dave Caulkins, 415-328-2411 (work), 415-948-5753 (home), 437 Mundel Way, Los Altos, CA 94022.

COMPUTER STORE SURVEY

An opinion survey of the nation's computer stores is available from Image Resource. The survey, which involved both questionnaires and telephone interviews, covers over 450 computer stores and shows how the stores rate the top 37 personal computer manufacturers on 17 performance parameters such as delivery, packaging, warranty, pricing and reliability. The report also covers business problems, trends, and market characteristics observed by the store managers. The survey will be published annually; the current survey is $495 to manufacturers, $295 to others.

Image Resource, 717 Lakefield Road, Suite B, Westlake Village, CA 91361. (805) 495-6277.

STUDENT LEARNING AID

DataMan, introduced by Texas Instruments, is a new calculator-based learning aid to help youngsters seven years old and up sharpen their math skills. Designed along the lines of the highly successful TI "Little Professor" introduced a year ago, it carries a higher degree of sophistication, more functions and extra features to provide youngsters considerable learning breadth and depth. Packaged in an imaginative design, DataMan offers many learning activities and intriguing math strategy games. A special "beat-the-clock" timing feature adds to the fun and challenge of numbers. Correct answers are rewarded with a "whiz-bang," highly visual action-packaged display on the order of modern stadium scoreboards.

DataMan can be adjusted to a youngster's achievement level so that learning experiences can be increased as the student gains confidence and skill.

This TI mathematics learning tool can be set for practice in any of four functions — addition, subtraction, multiplication or division. Among learning activities are: *Answer Checker, Problem Storage, Math Tables, Missing Numbers, Wipe-Out, Number Guesser,* and *Force Out.* $24.95, suggested retail.

Texas Instruments Inc., Inquiry Answering Service, P.O. Box 53 (Attn: DataMan), LubbockTX 79408.

FIRST WEST COAST COMPUTER FAIRE CONFERENCE PROCEEDINGS

If you missed the First West Coast Computer Faire, held in San Francisco this last April, here are all the papers presented at that show, also of interest to those who attended the show but who couldn't get to hear all the papers of interest, of which there were many. These 352 pages give you 90 papers, the banquet speeches, and a three-page list of all of the more than 170 exhibitors. $12.00, plus 0.68 for shipping, plus 6% sales tax for Californians.

Computer Faire, Box 1579, Palo Alto, CA 94302. (415) 851-7664.

LIST OF S-100 BOARDS

At irregular intervals, Robert Elliott Purser publishes a free list of almost all the S-100 boards available anywhere. Send a self-addressed envelope with a 13¢ stamp, or with two international reply coupons, to P.O. Box 466, El Dorado, CA 95623.

STAR TREK

STAR TREK INFORMATION

For a dollar, you get an amazing amount of information about almost anything to do with Star Trek, in the 15 double-sided pages of the "Yellow Pages of Star Trek," containing a listing of future conventions, three pages of books, a couple hundred clubs in 46 states and seven foreign countries, sale items offered offerings by individual fans, dealers, professionals, and manufacturers, and a couple of hundred fanzines. $1.00.

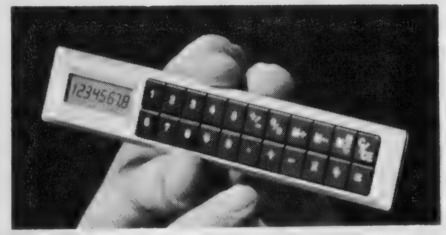
Star Trek Welcommittee, Allyson Whitfield, P.O. Box 206, New Rochelle, NY 10804.

STAR TREK BLOOPERS AND OUTTAKES

Star Trek blooper pictures and outtakes (Spock sucking a Tootsie Pop, etc). Also general pics. Send SASE for free catalog.

Eric Essman, 1344 Terry, Clovis, CA 93612.

CALCULATORS

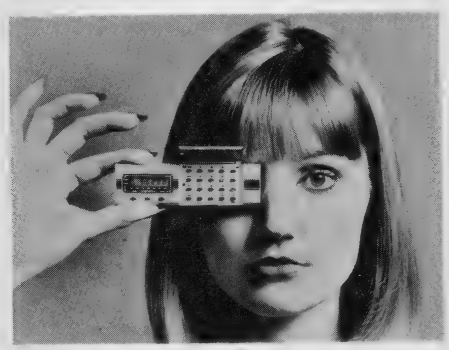


DATACLIP

A new carry-along calculator about the size of a six-inch ruler has been introduced by Texas Instruments. Called the DataClip, it has five functions, a bright eight-digit liquid crystal display, and operates up to 1000 hours on a set of batteries.

It is a little longer than a ballpoint and about an inch wide. The four basic functions are complemented by a memory that can be added to or subtracted from. Memory recall and clear memory can also be executed. The DataClip comes complete with batteries and attractive carrying case. Suggested retail price is $34.95.

Texas Instruments Inc., Inquiry Answering Service, P. O. Box 53 (Attn: DataClip), Lubbock, TX 79408.



CALCULATING CLOCK

A new mini "calculating clock," the Casio, MQ-1, is a little larger than a cigarette lighter. In addition to being a four-function (add, subtract, multiply and divide) calculator, the MQ-1 is also a timepiece. It shows the time in hours, minutes and seconds. It has a calendar feature that displays the year, month, date and day of the week. A stopwatch feature records time up to 23 hours, 59 minutes, 59.9 seconds. It shows each 1/10 of a second. $59.95

Casio, Inc., 15 Gardner Road, Fairfield, N.J. 07006. (201) 575-7400. ■

We're the memory house

Associated Electronics Company
1885 W. Commonwealth, Unit G
Fullerton, CA 92633
(714) 879 - 7541   879 - 7707

ORDER FORM

	QTY. KIT	QTY. ASSEMBLED
AE 8KS	☐	☐
AE 16 KPS	☐	☐
AE 1702A PRGM	☐	☐
BROCHURE	☐	

Name: ____________________

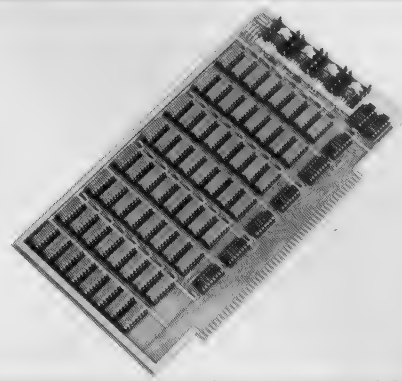
Address: ____________________

____

We accept: Checks • Money Orders • C.O.D. • Credit Cards
Master Charge # ____________
BankAmericard # ____________

AE 8 KS (8K STATIC RAM)

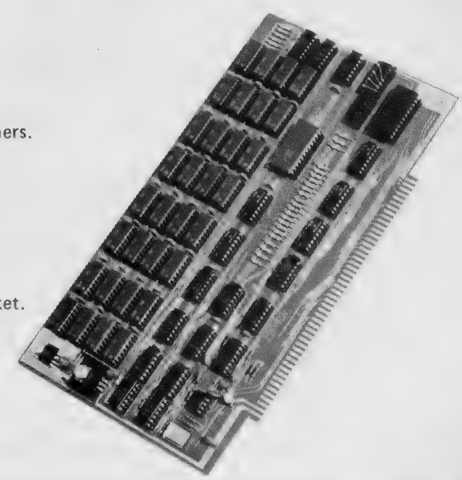
- 8192 bytes of static memory.
- Access time: 500 nanoseconds.
- Memory chips are Signetics 21L02's.
- Memory allocation is in 8K increments to any location (switch selectable).
- Memory protection (full 8K) selectable with a switch.
- Fully socketed and buffered.
- Solder mask and character screen for ease of assembly.



Kit $189.00   Assembled $289.00

AE 16 KPS (16K PSEUDO-STATIC RAM)

- Runs with all popular processors including Polymorphics, Sol, Chromemco, ZPU, and others.
- Runs with Micropolis, North Star, Tarbell and other mass storage systems.
- Full speed memory chips (500 nanoseconds) with no wait states.
- All memory management functions on board. No software management required.
- Low power consumption.... 2.5 to 5 watts maximum!!!
- Headers allow allocating memory in 4K blocks to ANY LOCATION within the 65K 8080 or Z80 memory space. Allocation must be on 4K boundaries.
- Phantom, memory protect-unprotect, and other functions jumper selectable.
- This memory card is as DMA compatible as any other pseudo-static memory on the market. The DMA controller MUST HONOR ALL PROCESSOR SIGNALS!!! ASK DYNABYTE which DMA controllers they run with... then ask us!!!!
- Fully socketed and buffered.



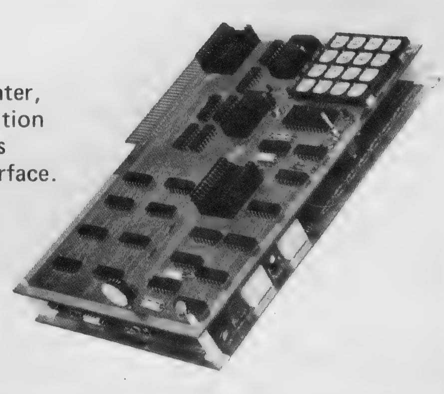
Kit $349.00   Assembled $449.00

AE 1702A PRGM (MANUAL 1702 PROGRAMMER)

Our new EPROM programmer has a vast array of improvements over the original design... with NO INCREASE IN PRICE!!! THE ORIGINAL DESIGN has sold to John F. Kennedy Space Flight Center, Jet Propulsion Labs, Los Alamos Scientific Lab, Stanford Linear Acceleration Center, IBM, XEROX, 3M, and many others. The new design incorporates on-board 3 state buffers and two 16 pin headers for direct processor interface.

The new design also incorporates a new keyboard for increased reliability and improved function selection.

Write for additional details.



Kit $189.00   Assembled $299.00

Programming Techniques: File Structures (Part 1)

John Lees

The primary use of general-purpose computers is data processing, and most of data processing is file handling. To make effective use of your computer, sooner or later you'll have to deal with files. So here's an introduction to the wide world of file structures.

To begin with, a few definitions: A *file* is an organized collection of related information. (A collection of files can be called a data base.) A file consists of *records*, all of which usually have the same basic structure. Each record can, in turn, be subdivided into *fields*, or *elements*.

A file can exist in memory, on paper tape, cassette, magnetic tape, disk, or any other type of storage. Storage can be considered in two categories, that allowing only sequential access (all kinds of tape and some primitive disk systems); and that allowing random access (semiconductor or core memory and most disk systems). Some tape systems purport to have random access, but random access on tape can only be achieved with considerable access time and space overhead.

Sequential File

The simplest and most common file structure is *physical sequential.* Records are arranged in some order, one after another in the file, and are physically accessed in that order. Tape files are by their very nature physically sequential.

Let's consider an example of a sequential file. Say you want to keep a catalog of all the books in your library, or all your record albums, musical scores, paintings, or any similar item. A record in such a file might look like this:

AUTHOR	TITLE	PUBLISHER	ADDRESS	BINDING	PAGES	PRICE	DATE

The file would consist of one such record for each book in your library. Probably the order in which you would choose to keep such a file stored would be alphabetical by the author field (which would be the *key field* for the file), but of course you could store it in any order you wish. The order you decide to use is important though, since you don't want to have to sort the file each time you use it.

Once you set up your library file, you'll want programs to add books or delete books and possibly to allow you to modify a record, although you could get by with deleting, then adding, to modify. You'll want the capability to print, say, all titles by one author, or by one publisher, or all hardbound books, or all books published in 1974.

Two Transports

If you're using tape, you'll need two transports to be able to keep the file in order, when you add or delete records. To do that, you'd write a program to take all the additions or deletions, sort them and keep them in memory. Then your program would read from one tape, writing records out to the other tape until it reaches the point for an add or delete. It would do the add or delete and then keep on reading/writing until the entire file had been processed.

You may have noticed that a lot of space is being wasted in our sample file. There is only a small number of different publishers, yet that information is repeated in every record. Very wasteful! To save space, we could create two files, one file consisting of a modification of the records we already have, with the publisher information replaced with a code:

AUTHOR	TITLE	P-CODE	BINDING	PAGES	PRICE	DATE

and another file containing the publisher information:

P-CODE	PUBLISHER	ADDRESS

These two files could be used in this way. Put the Publisher file first on the tape, followed by the Library file. As the first step in using the Library file, read the Publisher file into the memory since it is relatively small. Now, as records are read from the Library file, the publisher code can be matched up with one in the Publisher file and the information in the record in the Publisher file used to print the book listing. This look-up will be fast since it is done in memory. The small amount of extra processing time used is well justified by the savings in file space. This same principle can be applied whenever a field contains often-repeated information. Of course the information must be longer than the code used to replace it. It wouldn't pay to do this with the date field, for instance.

Tight Space

If you're real hard-up for space, the leading "1" need not actually be stored in the date field. Similarly, don't store a "$" or even the decimal point in the price field. Have the program add them when it adds the publisher information to the book listing being printed. You can save even more bytes by storing the price and date in binary and converting. But don't get carried away if you don't need to save the space.

All well and good, but what if you have collections, such as science-fiction anthologies, and want to be able

to find authors and titles of stories in the collections? You could do about the same thing we did with publisher information. Add a code to the author field:

AUTHOR	B-CODE	TITLE	P-CODE	BINDING	PAGES	PRICE	DATE

This new code could mean if 0 then the book is not a collection, else the code would match up with a set of records in a short-story file which would give the contents of that collection.

But that isn't really what you want. That scheme will let you list the contents of a collection, but is of no help in finding out if you have in your library a short story that only appears in a collection. What to do? You could, instead of having a separate short-story file, include these records in the main file. Avoiding redundant information, you would now have a file consisting of three different types of records:

TYPE 0	AUTHOR	TITLE	P-CODE	BINDING	PAGES	PRICE	DATE

TYPE 1	B-CODE	AUTHOR	TITLE	P-CODE	BINDING

TYPE 2	B-CODE	AUTHOR	TITLE

A type 0 record would be a normal book. A type 1 record would be a collection and such a record would contain an additional field with a book code, which would match a book code in type 2 records containing the authors and titles of the short stories in the collection. To make this file easily usable with a sequential storage medium, you'd probably want to group all the records together by types (almost, in effect, giving three files), in alphabetical order by author within type.

Now if you want to see if K is in your library, the program would look for a record of any type with K in the author field. If the record(s) found with K are of type 2, then the program would also look for a type 1 record with matching b-code and tell you what collection the story by K appears in. If you think that this kind of thing could take a very long time on a cassette, you're perfectly correct. But what's your hurry?

Faster, Faster

Well, maybe you're writing this system for the school library and you have a legitimate reason for wanting the search for a book to take less than half an hour. Hopefully you can get a disk or two, because you've exceeded the capabilities of a sequential-file structure. The rest of the structures we're going to discuss require random-access devices.

The drawbacks to the plain old sequential file are obvious. If you're on Heinlein, you know that Vonnegut is somewhere further on and that you've passed Ellison. Vonnegut you'll eventually come to, but Ellison can only be reached by going back to the beginning and starting over again. An unhandy state of affairs if speed is of any importance at all.

Index

So, enter the next bright idea in file structures, the index. Although the combination of indexes and sequential files will not, as IBM once tried to convince the world, solve all problems, it does help a little. Imagine a dictionary with no way of telling where each letter begins and you'll quickly appreciate the utility of an index. The idea is the same with a file. You have the master file and an *index file* which contains the information on where certain categories begin in the master file. This could be in terms of record number, memory address, disk sector, or (shudder) tape block. Now if you want to find Heinlein, the program looks in the index file and goes right to the beginning of the H's. This is of limited use on tape, since you still have to move all that tape past the read head slowly enough to count blocks.

With our example, you could also have an index to help find the groups of short-story records and even the records for the collections themselves. So your program could go right to the record or group of records. A diagram for such a file system might look like this:

Alphabetical Index

A	LOCATION
•	
•	
•	
Z	LOCATION

Short Story / Collections Index

B-CODE	LOCATIONS OF SHORT STORY RECORDS	LOCATION OF COLLECTION RECORD
•		
•		
•		
B-CODE	LOCATIONS OF SHORT STORY RECORDS	LOCATION OF COLLECTION RECORD

Library Master File

TYPE 0	AUTHOR	TITLE	P-CODE	BINDING	PAGES	PRICE	DATE

TYPE 1	B-CODE	AUTHOR	TITLE	P-CODE	BINDING	PAGES	PRICE	DATE

TYPE 2	B-CODE	AUTHOR	TITLE

Publisher File

P-CODE	PUBLISHER	ADDRESS

Once you've figured out that conglomeration, you'll see that it saves a lot of work, at the expense of a little storage space. Using the indexes may speed things up, but a lot of sequential processing is still required, and the records within groups must still be kept in alphabetical order, thus requiring a lot of insert overhead. (Deletes are simple. Just adopt the convention that a type 3 record isn't there and so mark "deleted" records, every once in a while collecting the garbage and squishing things together.) Also, there are a couple of little bugs in that file system and a very high maintenance cost associated with updating all those index records if any of the Master file records are moved, as they will be each time an insert is performed.

It is possible to get away entirely from any reliance on sequential ordering, at the expense of a little more storage and a little more processing time. But processing time is cheap and maintaining a sequential file is a nightmare when you don't have much memory. So let's move on into the realm of list structures, linked lists, rings, trees, hierarchical files and such things. You ain't seen nothing yet! ■

MICROCOMPUTER

MICROPROCESSOR's

F-8	19.95
Z-80	25.00
Z-80A	35.00
CDP1802CD	24.95
AM2901	22.95
6502	12.95
6800	19.95
8008-1	8.75
8080A	15.95
TMS-9900TL	89.95

6800 SUPPORT

6810P	4.95
6820P	8.00
6828P	11.25
6834P	16.95
6850P	9.95
6852P	11.95
6860P	14.95
6862P	17.95
6880P	2.70

Z80 SUPPORT DEVICES

3881	12.95
3882	12.95

F-8 SUPPORT DEVICES

3851	14.95
3853	14.95

DYNAMIC RAMS

1103	1.50
2104	4.50
2107A	3.75
2107B	4.50
2107B-4	4.00
TMS4050	4.50
TMS4060	4.50
4096	4.50
4116	42.00
MM5270	5.00
MM5280	6.00
MCM6605	6.00

CHARACTER GENERATORS

2513 UP	6.75
2513 DOWN	6.75
2513 UP (5v)	9.95
2513 DOWN(5v)	10.95
MCM6571	10.80
MCM6571A	10.80
MCM6572	10.80
MCM6574	14.75
MCM6575	14.75

PROM'S

1702A	5.00
2704	15.00
2708	20.00
2716	38.00
3601	4.50
5203AQ	4.00
5204AQ	6.00
6834	16.95
6834-1	14.95
82S23B	4.00
82S129B	4.25
8223B	2.70

8080A SUPPORT DEVICES

8212	3.95
8214	9.95
8216	4.50
8224	4.95
8228	8.75
8238	8.00
8251	12.00
8253	28.00
8255	12.00
8257	25.00
8259	25.00

STATIC RAMS	1-24	25-99	100
21L02 (450)	1.50	1.40	1.25
21L02 (250)	1.95	1.80	1.50
21L11	4.25	4.10	3.95
1101A	1.49	1.29	1.10
2101-1	2.95	2.75	2.60
2102	1.25	1.15	1.00
2102-1	1.50	1.30	1.15
2111-1	4.00	3.50	3.25
2112-1	3.00	2.80	2.69
2114	17.95	16.95	16.50
31L01	2.50	2.35	2.00
3107	3.95	3.70	3.25
4200A	12.95	12.50	11.95
4804/2114	17.95	16.95	16.50
5101C-E	11.95	11.25	10.25
74C89	3.25	3.05	2.85
74S201	4.50	4.30	4.25
7489	2.25	2.10	1.90
8599	1.88	1.75	1.60
9102BPC	1.65	1.45	1.30

KEYBOARD CHIPS

AY5-2376	13.95
AY5-3600	13.95

UART'S

AY5-1013A	5.50
AY5-1014A	8.95
TR-1602A	5.50
TMS-6011	6.95
IM-6402	10.80
IM-6403	10.80

FLOPPY DISC CONTROLLER

1771B	55.95
1771B-01	59.95

SHIFT REGISTERS STATIC

2518B	3.95
2533V	2.00
TMS3002	1.00
TMS3112	3.95
MM5058	2.00

KIM

KIM-1	245.00
6502	12.95
6520	9.00
6522	9.25
6530-002	15.95
6530-003	15.95
6530-004	15.95
6530-005	15.95

USRT

S-2350	10.95
WD1671B	29.95

WAVEFORM GENERATOR

8038	4.00
MC4024	2.50
566	1.75

MISC. OTHER COMPONENTS

NH0025CN	1.70	P-3408A	5.00
NH0026CN	2.50	P-4201	4.95
N8T20	3.50	MM-5320	7.50
N8T26	2.45	MM-5369	1.90
74367	.90	DM-8130	2.90
DM8098	.90	DM8131	2.75
1488	1.95	DM-8831	2.50
1489	1.95	DM-8833	2.50
D-3207A	2.00	DM-8835	2.50
C-3404	3.95	SN74LS367	.90
		SN74LS368	.90

TV Game Chip
TMS1955NL
Now Only $10.95

PerSci DISK AND CONTROLLER

Use the PerSci Disk and Controller now with the Info 2000 Adapter for the S-100 Bus.

INFO 2000 "SPECIAL" (includes Model 277 Dual Drive, Model 1070 Controller, Case with power supply and fan, and cable)	$2,150
Model 277 Dual Diskette Drive	$1,130
Model 1070 Controller	$740
Slimline case with power supply and fan	$280
Adapter for the S-100 Bus (Kit)	$120

JADE PARALLEL/SERIAL INTERFACE KIT
$124.95 KIT

JADE VIDEO INTERFACE KIT
$89.95 KIT

8K STATIC RAM BOARD

ASSEMBLED & TESTED

250ns.	$199.95
350ns.	$189.95
450ns.	$169.95

* WILL WORK WITH NO FRONT PANEL
* FULL DOCUMENTATION
* FULLY BUFFERED
* S100 DESIGN
* ADEQUATELY BYPASSED
* LOW POWER SCHOTTKY SUPPORT IC'S

KIT

250ns.	$169.95
350ns.	$139.95
450ns.	$129.95

IMSAI/ALTAIR **S-100** COMPATIBLE

JADE Z80 KIT

—with PROVISIONS for ONBOARD 2708 and POWER ON JUMP

$135.00 EA. (2MHZ)

$149.95 EA. (4MHZ)

BARE BOARD $35.00

IMSAI/ALTAIR **S-100** COMPATIBLE

JADE 8080A KIT

– WITH EIGHT LEVEL VECTOR INTERRUPT

$110.00 KIT

BARE BOARD $35.00

JADE
Computer Products
5351 WEST 144th STREET
LAWNDALE, CALIFORNIA 90260
(213) 679-3313

RETAIL STORE HOURS M-F 9-7 SAT. 9-5

Discounts available at OEM quantities. Add $1.25 for shipping. California residents add 6% sales tax.

CATALOG FREE WITH $10.00 ORDER

Design An LP Record File-Storage System

Readers of *Creative Computing* are invited to design and implement a pragmatic, workable data-base system to meet the design specifications given below. The writeup must completely describe hardware (preferably off-the-shelf), systems software (operating system, language compiler or interpreter, etc.), and applications software (including complete listings and sample runs). The best design received by March 1, 1978 will be published in the May/Jun 1978 *Creative Computing* and will receive a $50 bonus in addition to payment for the article at our regular rate.

Summary: The problem is to design a computer system (hardware, software, etc.) to store and retrieve information in support of a typical home record library. The system must be able to easily do the following: (1) allow updating as new records are purchased; (2) permit retrieval of listings defined by primary category; such as all classical; by secondary category, such as all Bach, or all Columbia, or all piano concerto; or by any combination of up to three variables, such as all Shostakovich symphonies on Melodiya.

Problem Statement: The LP record library is assumed to have the following characteristics at present.

Number of Records	*Primary Category*	*Secondary Categories*	*Examples*
600	Classical	Composer	J.S.Bach
		Composition	Cantata #131
		Subcategory	Choral
		Label and Number	MHS 760
		Add'l Compositions	Bach: Cantata 149
150	Popular	Artist	Reddy, Helen
		Title	Music, Music
		Label and Number	Cap ST-11547
30	Folk	Artist	Theodorakis, Mikis
		Title	Hello Greece!
		Label and Number	Minerva 22030
50	Shows, Movies	Title	Where Eagles Dare
		Subcategory	Soundtrack
		Label and Number	MGM 2315 036
20	Children's	Title	Snow White
		Artist	Soundtrack
		Label and Number	Disney 1201
20	Band, Spoken, Misc.	Title	Sounds of Auto Racing
		Subcategory	Sound Effects
		Label and Number	Fleetwood FLD-1-S

Information in the computer should, as much as possible, be kept compatible with the Schwann record catalogs. In the classical category, "additional compositions" should allow for up to 6 additional compositions by the same or other composers. More than six compositions and the record or album can probably be classified a collection and the additional compositions block might say, for example, "Berlioz: 5 other overtures."

Preceding the label and number should be an indication of the number of records if it is a multiple record set; for example 2-Angel 3573. This is consistent with Schwann.

Classical subcategories are: ballet, opera or operetta, piano concerto, violin concerto, other concerto, sonata or rondo, organ, solo piano, solo other instrument, chamber (quintet, quartet, trio, etc.), choral, symphony or sinfonia, orchestral, misc. other.

Shows and movies subcategories are: soundtrack, original cast, orchestral version, other. Band, spoken, misc. subcategories are: band, spoken, demonstration, test, sound effects, misc.

The Schwann catalog should give a good idea of the required length of each field. As a rough guide to the classical ones.

Category	*Field Length*
Composer	24
Title	60
Subcategory	2 (abbreviated)
Label and Number	20 (using abbreviations)
Add'l compositions	180 (seldom will all this be required)

Another way of providing for "add'l compositions" is to reference a pointer which automatically retrieves the information from the other heading and prints it out. In other words, the example of MHS 760 above which lists Bach: Cantata 149 as "add'l compositions" would have a pointer value, say 182, which would go to Composer/-Title #182 which, of course, is J.S. Bach Cantata 149.

Other Considerations: Non-standard hardware and systems software should be avoided if at all possible. Hard copy is desirable but not required.

It is desirable to make the system as general as possible. Thus it could be used for storage and retrieval of information on a book library, stamp collection, coin collection, comic-book collection, antique auto parts, photo-negative file, recipe file, or any other similar collection.

Be creative but avoid overkill! — DHA ■

Computer Professionals' Book Club

MEMBERSHIP ORDER CARD

Please enroll me as a member and send me the three books indicated. I am to receive the two bonus books at the introductory price of $2.95 plus my first selection at the discounted price for members, plus tax, postage, and handling. If not completely satisfied, I may return the books within 10 days and request that my membership be cancelled. If I keep the books, I agree to take a minimum of three additional books of my own chosing during the next two years. I will receive the Club Bulletin 13 times a year. If I want to examine the featured selection, I need take no action. It will be shipped automatically. If, however, I want an alternate selection—or no book at all—I simply notify the Club by returning the convenient card always enclosed. I will always have a minimum of 10 days in which to return the card, and you will credit my account fully, including postage, if this is not the case. Membership in the Club is continuous but cancellable by me at any time after the four-book purchase requirement has been filled. *This order subject to acceptance by McGraw-Hill. Orders from outside the continental U.S. must be prepaid. All prices subject to change without notice. Company, business, or institutional tax exemption status is not applicable to purchases made through individual Club memberships. Offer good for new members only.*

INDICATE BY NUMBER THE THREE BOOKS YOU WANT

Write Code # of 1st Bonus Book Here	Write Code # of 2nd Bonus Book Here	Write Code # of First Selection Here

SEND NO MONEY. SIMPLY MAIL THIS CARD TODAY AND WE WILL BILL YOU.*

Name________________

Street Address________________

City________________

State__________ Zip________

***EXTRA SAVINGS:** Remit in full with your order, plus any local and state tax, and McGraw-Hill will pay all postage and handling charges.

CREATIVE COMPUTING **P39266**

Computer Professionals' Book Club

MEMBERSHIP ORDER CARD

Please enroll me as a member and send me the three books indicated. I am to receive the two bonus books at the introductory price of $2.95 plus my first selection at the discounted price for members, plus tax, postage, and handling. If not completely satisfied, I may return the books within 10 days and request that my membership be cancelled. If I keep the books, I agree to take a minimum of three additional books of my own chosing during the next two years. I will receive the Club Bulletin 13 times a year. If I want to examine the featured selection, I need take no action. It will be shipped automatically. If, however, I want an alternate selection—or no book at all—I simply notify the Club by returning the convenient card always enclosed. I will always have a minimum of 10 days in which to return the card, and you will credit my account fully, including postage, if this is not the case. Membership in the Club is continuous but cancellable by me at any time after the four-book purchase requirement has been filled. *This order subject to acceptance by McGraw-Hill. Orders from outside the continental U.S. must be prepaid. All prices subject to change without notice. Company, business, or institutional tax exemption status is not applicable to purchases made through individual Club memberships. Offer good for new members only.*

INDICATE BY NUMBER THE THREE BOOKS YOU WANT

Write Code # of 1st Bonus Book Here	Write Code # of 2nd Bonus Book Here	Write Code # of First Selection Here

SEND NO MONEY. SIMPLY MAIL THIS CARD TODAY AND WE WILL BILL YOU.*

Name________________

Street Address________________

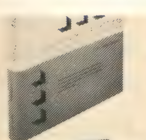
City________________

State__________ Zip________

***EXTRA SAVINGS:** Remit in full with your order, plus any local and state tax, and McGraw-Hill will pay all postage and handling charges.

CREATIVE COMPUTING

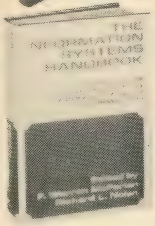
P39267



Implementation and Applications
by C. Weitzman
Pub. price, $19.50
Club price, $14.50

for $2.95 with your first club selection

Be sure to consider these books as well—



769/29X
THE INFORMATION SYSTEMS HANDBOOK
by F. W. McFarlan & R. L. Nolan
Pub. price, $30.00
Club price, $22.50

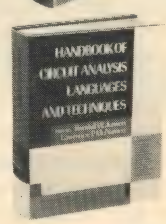
767/009
DIGITAL SIGNAL PROCESSING
by A. V. Oppenheim & R. W. Schafer
Pub. price, $22.95
Club price, $17.95

768/03X
MICROPROGRAMMABLE COMPUTER ARCHITECTURES
by A. B. Salisbury
Pub. price, $13.50
Club price, $11.40

768/749
MICROPROCESS/MICROPROGRAMMING HANDBOOK
by B. Ward
Pub. price, $9.95
Club price, $8.45

769/419
MODERN GUIDE TO DIGITAL LOGIC: PROCESSORS, MEMORIES & INTERFACES
by United Technical Publications, Inc.
Pub. price, $9.95
Club price, $8.45

by The Diebold Group, Inc.
Pub. price, $29.50
Club price, $21.00



769/656
HANDBOOK OF CIRCUIT ANALYSIS LANGUAGES AND TECHNIQUES
by R. Jensen & L. McNamee
Pub. price, $34.15
Club price, $28.50

save time and money
by joining McGraw-Hill's new

COMPUTER PROFESSIONALS' BOOK CLUB

THIS new professional club is designed to meet your day-to-d— on-the-job needs by providing practical books in your field on a regular basis at *below publisher prices.* If you're missing out on important technical literature—if today's high cost of reading curbs the growth of your library—here's the solution to your problem.

The Computer Professionals' Book Club was organized for you, to provide an economical reading program that cannot fail to be of value. Administered by the McGraw-Hill Book Company, all books are chosen by qualified editors and consultants. Their understanding of the standards and values of the literature in your field guarantees the appropriateness of the selections.

How the Club operates: Every month you receive free of charge *The Computer Professionals' Book Club Bulletin.* This announces and describes the Club's featured book of the month as well as alternate selections available at special members' prices. If you want to examine the Club's feature of the month, you do nothing. If you prefer one of the alternate selections—or if you want no book at all—you notify the Club by returning the card enclosed with each *Bulletin.*

As a Club Member, you agree only to the purchase of four books (including your first selection) over a two-year period. Considering the many books published annually, there will surely be at least four you would want to own anyway. By joining the club, you save both money and the trouble of searching for the best books.

MAIL THIS COUPON TODAY

COMPUTER PROFESSIONALS/Book Club
P.O. Box 582 Princeton Road, Hightstown, New Jersey 08520

Please enroll me as a member and send me the three books indicated. I am to receive the two bonus books at the introductory price of $2.95 plus my first selection, plus tax, postage, and handling. If not completely satisfied, I may return the books within 10 days and request that my membership be cancelled. If I keep the books, I agree to take a minimum of three additional books during the next two years at special Club prices (guaranteed 15% discount, often more). I will receive the Club Bulletin 12 times a year. If I want to examine the featured selection, I need take no action. It will be shipped automatically. If, however, I want an alternate selection—or no book at all—I simply notify the Club by returning the convenient card always enclosed. I will always have a minimum of 10 days in which to return the card and you will credit my account fully, including postage, if this is not the case. Membership in the club is continuous but cancellable by me at any time after the four-book purchase requirement has been filled. *This order subject to acceptance by McGraw-Hill. Orders from outside the continental U.S. must be prepaid. Company, business, or institutional tax exemption status is not applicable to purchases made through individual Club memberships. All prices subject to change without notice. Offer good for new members only.*

Write Code # of 1st *bonus* book selection here	Write Code # of 2nd *bonus* book selection here	Write Code # of 1st book selection here

Name________________

Address________________

City________ State______ Zip______

EXTRA SAVINGS: Remit in full with your order, plus any local and state tax, and McGraw-Hill will pay all regular postage and handling charges. **P39265**

Business Reply Mail
No Postage Stamp Necessary if Mailed in the United States
Postage will be paid by
Computer Professionals' Book Club
P.O. Box 582
Hightstown, New Jersey 08520

FIRST CLASS
PERMIT NO. 42
Hightstown, N. J.

system must be able to easily do the following: (1) allow updating as new records are purchased; (2) permit retrieval of listings defined by primary category; such as all classical; by secondary category, such as all Bach, or all Columbia, or all piano concerto; or by any combination of up to three variables, such as all Shostakovich symphonies on Melodiya.

Problem Statement: The LP record library is assumed to have the following characteristics at present.

Number of Records	*Primary Category*	*Secondary Categories*	*Examples*
600	Classical	Composer	J.S.Bach
		Composition	Cantata #131
		Subcategory	Choral
		Label and Number	MHS 760
		Add'l Compositions	Bach: Cantata 149
150	Popular	Artist	Reddy, Helen
		Title	Music, Music
		Label and Number	Cap ST-11547
30	Folk	Artist	Theodorakis, Mikis
		Title	Hello Greece!
		Label and Number	Minerva 22030
50	Shows, Movies	Title	Where Eagles Dare
		Subcategory	Soundtrack
		Label and Number	MGM 2315 036
20	Children's	Title	Snow White
		Artist	Soundtrack
		Label and Number	Disney 1201
20	Band, Spoken, Misc.	Title	Sounds of Auto Racing
		Subcategory	Sound Effects
		Label and Number	Fleetwood FLD-1-S

Information in the computer should, as much as possible, be kept compatible with the Schwann record catalogs. In the classical category, "additional compositions" should allow for up to 6 additional compositions by the same or other composers. More than six or rondo, organ, solo piano, solo other instrument, chamber (quintet, quartet, trio, etc.), choral, symphony or sinfonia, orchestral, misc. other.

Shows and movies subcategories are: soundtrack, original cast, orchestral version, other. Band, spoken, misc. subcategories are: band, spoken, demonstration, test, sound effects, misc.

The Schwann catalog should give a good idea of the required length of each field. As a rough guide to the classical ones.

Category	*Field Length*
Composer	24
Title	60
Subcategory	2 (abbreviated)
Label and Number	20 (using abbreviations)
Add'l compositions	180 (seldom will all this be required)

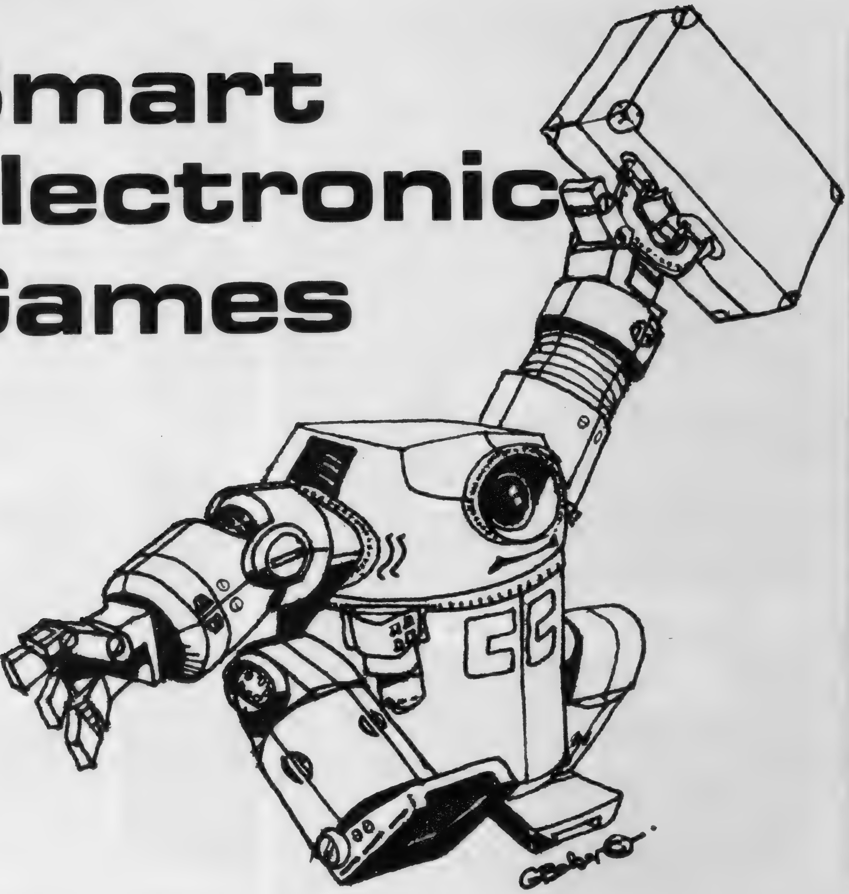
Another way of providing for "add'l compositions" is to reference a pointer which automatically retrieves the information from the other heading and prints it out. In other words, the example of MHS 760 above which lists Bach: Cantata 149 as "add'l compositions" would have a pointer value, say 182, which would go to Composer/-Title #182 which, of course, is J.S. Bach Cantata 149.

Other Considerations: Non-standard hardware and systems software should be avoided if at all possible. Hard copy is desirable but not required.

It is desirable to make the system as general as possible. Thus it could be used for storage and retrieval of information on a book library, stamp collection, coin collection, comic-book collection, antique auto parts, photo-negative file, recipe file, or any other similar collection.

Be creative but avoid overkill! — DHA ■

Smart Electronic Games



As anyone who has visited a department store or read a mail-order catalog recently knows, recreation and leisure-time options are rapidly expanding. One little corner of this market, perhaps the fastest-growing single segment, incorporates solid-state electronics into video games, stand-alone games, and calculator variants. In the past *Creative* has taken some brief looks at video games ("Playing Pong to Win," May/Jun 75; "Odyssey Video Games," Nov/Dec 76). Over the next several issues we'll be taking an in-depth look into the mysteries of a number of the new stand-alone games (see the article on Comp IV elsewhere in this issue).

For those of you interested in what's commercially available for gift-giving this holiday season, here is a roundup (undoubtedly incomplete) of some of the new smart electronic games. The games are widely discounted, but not nearly to the extent of hand-held calculators. Also, we found some stores selling them at higher-than-list prices! A word of warning: quality control may not be up to snuff — one of the games we obtained quit in the first 5 minutes. But most worked OK and were generally lots of fun! — DHA

Mattel Auto Race

Sliding control steers car (bright light blip) to avoid hitting computer-controlled oncoming cars (dimmer light blips). At end of lap, car will begin new lap. Gear-shift controls speed. Simulated engine sounds become louder as speed increases. Elapsed time shown. On collision, beep sounds and your car is pushed back a space or two. Complete the 4 laps in 99 seconds or less, you win and a pulsating beep is emitted. If you fail to do so, clock stops at 99; constant beep is heard.

Beige plastic case 1x3x5 in. Uses one 9-volt battery. Retail range, $20 to $25.

Mattel Football

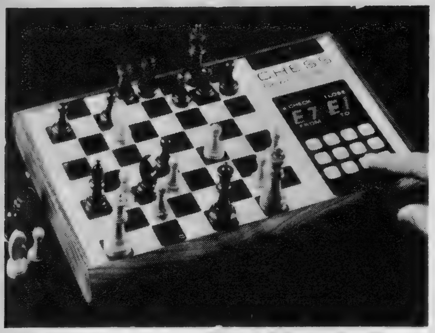
You control running back (bright light blip) through the computer-controlled defensive tacklers (dimmer light blips). If running back hits a tackler, a simulated whistle sounds. ST (status) key shows new down, yard line and no. of yards for first down. SC (score) key reveals both teams' scores and time remaining. On scoring, you hear the "victory charge." K (kick) key is used for punting and field goals. Game time is 10 min. Pro 1/Pro 2 switch allows you to choose average or expert opposition.

Beige plastic case, 1x3x5 in. Uses one 9-volt battery. Retail range, $30 to $35.

Mattel Missile Attack

It's you against the computer as you defend your city! The computer launches missiles aimed to destroy your city, and you launch anti-missile missiles to intercept and destroy the incoming barrage. The more you destroy, and the higher in the sky you do so, the more points you get. If you miscalculate and an attacking missile makes a direct hit on your city, the computer plays a mini rendition of "Taps"!

Blue-grey plastic case, 1x3x5 in. Uses one 9-volt battery. Retail range, $20 to $25.



Fidelity Electronics Chess Challenger

The board is defined by numbers along the X-axis and letters on the Y-axis. Moves are entered on the calculator-like keyboard by defining the current location of a piece and where you want to move it to. You may castle or capture en passant, although the computer isn't programmed to use the latter move. Within seconds of your move, the computer analyzes possible counter-moves and responds with its best choice. It can be beaten by an average player from 25% to 75% of the time.

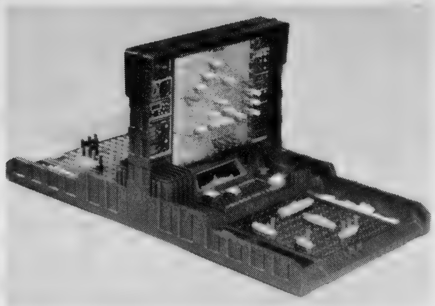
Experts who want to play a more difficult game can send their unit, along with $75, to the manufacturer for upgrading with a more complex ROM, which increases the difficulty level, has three levels of play. A newer model, which has three levels of play and is essentially the same as an upgraded basic model, but with some cosmetic changes, is $275.

Walnut wood housing, 12x8x1 in. Operates on line current. Retail range, $160 to $200.

Milton Bradley Comp IV

Using the calculator pad, you try to guess a secret 3-digit number (like Bagels), 4-digit one (Mastermind) or 5-digit one. Comp IV tells you how many digits are correct and how many are in the correct position. Clever deduction will allow you to get the number in the fewest possible guesses. Comp IV prods you with flashing lights if you delay too long. Use a different number each round, or the same one if two or more players are competing against each other.

Plastic console, 7½x4x4 in. Uses one 9-volt battery. Retail range, $20 to $40.



Milton Bradley Electronic Battleship

Two players, acting as fleet commanders, determine the best locations for each of their ships in this game of electronic hide-and-seek on the high seas. The continuous sound of sonar adds realism to the mission, as you set your console controls to fire on sectors of the board where the opponent's ships may be hidden. Press the Fire button, and you hear the whoosh of a torpedo heading for its target; hits are indicated by a flash of light on the scope and the sound of a realistic explosion. When you've sunk your opponent's entire fleet, Electronic Battleship sounds a continuous whoop of victory.

Plastic board and display, 21x10¾x9 in. Uses four 9-volt batteries. Retail range, $30 to $50.

Parker Brothers Code Name: Sector

The human players (1 to 4) are destroyer commanders, moving their ships among 4800 grids on a nautical chart. The microprocessor stores, computes, and displays speed, direction, and location of each ship. It also moves the sub, keeping its location secret but giving information that can be used to track and attack it. But be careful — that sub can torpedo destroyers! There are eleven control buttons. Comes complete with 8 replica subs for scoring, facsimile Navy parallel rule, tracking crayons, wiper, and sharpener. Not a simple game! Age 12 and up.

Plastic board and display, 19x13½x4 in. Uses one 9-volt battery. Retail range, $30 to $45.

Unisonic Vegas 21

Casino blackjack with full Las Vegas rules. Your initial stake can be up to $50 million; before each hand you designate your wager. Your winnings and losses are posted after each hand. You can make insurance bets, split pairs, double down. Automatic deck shuffle after 41st card is dealt. Doubles as a standard calculator.

Desk-top version woodgrain and gray plastic, 7x9x4 in. Large L.E.D. readout. Uses 3 C-cells. Retail range, $39 to $59.

Hand-held version chrome and black plastic, 1x3x5 in. Uses 3 AA batteries. Retail range, $30 to $50.

APF Mathemagician

A teaching calculator and device for game-playing all in one. Simple or complex 1- or 2-digit problems can be performed. Four-function calculator does multiplication, division, addition and subtraction. In addition, six games can be played — Gooey Gumdrop, The Number Machine, Walk-the-Plank, Football, Lunar Lander and Countin' On, each with its own plastic overlay. All games can be played by one or two people. In "Lunar Lander," you take over the controls of a lunar landing module at an altitude of 300 ft. above the moon's furface. Try and land the module safely! Switch to "Gooey Gumdrop" and try to find the secret location of the gumdrop bomb before it explodes. In "Walk the Plank," you try to guess the machine's secret number in three tries. If you lose, you walk the plank!

Black and gray plastic case, 8½x5x1½ in. Uses one 9-volt battery. $39.95. ■

Profile of a Smart Electronic Game . . .

Guess the random number generated by this new game, which gives hints.

COMP IV

Stephen B. Gray

"I am programmed to beat you," says the ad, and continues, in smaller print, "You are mortal. I am the product of millions of dollars of research. You will attempt to deduce the numbers I have hidden in my computer memory. I will reveal only which of your digits are correct and how many are in the right order. As your excitement mounts, you will get (ugh!) emotional. I will not. You are mortal. I am Comp IV, the new electronic challenge game with 32,000 number combinations, from Milton Bradley."

Description

Despite the provocative challenge, Comp IV looks quite simple and harmless. Weighing only about eleven ounces and fitting comfortably into one hand, it consists outwardly of no more than a blue plastic case with a 12-button keyboard, two vertical rows of five LEDs (light-emitting diodes) on a sloping display panel, an on/off switch, and a compartment for a nine-volt battery.

Inside, Comp IV looks almost as simple. Mounted under the 12 keys is a very thin Texas Instruments X-Y matrix on a plastic substrate. Pressing a key causes an X (row) wire to touch a Y (column) wire. The matrix is separated from the 12 keys by a thin sheet of plastic foam and a sheet of Mylar, mostly to protect the keyswitch matrix from anything that small (or large) hands might spill onto the keyboard.

The only other item concealed inside the case is the brains of the game, a 28-pin IC, also from TI, a one-chip microcomputer in the TMS1000 series. This family of ICs is used in every one of the TI calculators, as well as in another Milton Bradley game, Electronic Battleship, which will be reviewed at a later date

The TMS1000 series of microcomputers sells for less than $3 in quantity, and is recommended by TI for use in coin changers, fuel metering, oven control, games, automatic telephone answering, and many other applications. Each member of the family contains registers, a program counter, an arithmetic logic unit, RAM and ROM memory, I/O circuitry, and a clock. The particular one used in Comp IV contains 1,024 8-bit words of ROM memory for the random-number generator and the game-play algorithm, plus 64 4-bit RAM words to hold the numbers entered from the keyboard.

Basic Operation

After turning on Comp IV, you can check out the circuits by pressing the keys in the sequence 7-E-1-2-3-4-E (the 12 keys are labelled 0 through 9, plus R for Reset and E for Enter). If the unit is working correctly, the five lights in the left vertical row (Number) will flash along with the four bottom lights in the right vertical row (Sequence). The top right LED is the Ready indicator.

Press R (with or without having previously gone through the checkout procedure), and the LEDs will flash in seven preset combinations (the manual says Comp IV is scanning while it selects a number, but no doubt the number is selected long before the first combination of lights goes on. The flashing-light combinations simply make it look more like a computer). Then the Ready light goes on, to indicate that Comp IV is ready to accept your guess as to what number it has generated. In the numbers generated by Comp IV, no digit is repeated, so that the secret number will never be 553, for instance. That would make the game just a little too difficult for most people.

Your first entry determines the complexity of that particular game. If you wish to play a three-digit game, make your first guess a three-digit one, and enter it into the computer by pressing the E key. If you wish to play a four-digit or a five-digit game, make a first guess of four or five digits, before pressing E.

When you enter your guess, Comp IV compares it with the random number it has generated, and responds with lights on the sloping screen. The left row, marked NUMBER, indicates how many digits you guessed correctly; the right row, labelled SEQUENCE, shows how many of the digits guessed correctly are in the correct order (but not *which* numbers are in the right order).

As further entries are made, the display will help you figure out which digits to eliminate and which to confirm. If there is no match between your entry and Comp IV's number, the only light on the display will be the Ready light. If you enter too many or too few digits, the 1 NUMBER light will blink.

When you finally enter the exact number that Comp IV is holding, all the lights on the screen will flash (except the Ready light). To play another game, press R, wait for the seven sets of flashes, enter the number of digits you wish to play, and thus you start a new game against a new number generated by Comp IV.

A pad is provided for keeping track of your entries, as well as of digits you're sure are (or are not) in the hidden number.

Demonstration

Suppose Comp IV has generated 436. If your guess is 423, you'll get a 2 light in the NUMBER column because you got the 4 and 3 right, and a 1 light in the SEQUENCE column because you guessed the 4 in the right place. If your guess had been 432, you'd have a 2 light in the SEQUENCE column.

Extra Features

Comp IV has two features designed to keep you from dawdling. The R light "also serves as a timer to help you keep track of your game." After about 30 seconds, it will flash slowly, and after another 30 seconds, will begin to flash faster, and stay that way. Also, responses are displayed for only about 30 seconds, after which they go off and the ready light comes on.

But these features are only to remind you that time is passing; if you're playing alone, you can take all day to figure out Comp IV's hidden number. This is entirely unlike some of the other games that can be played alone (and which we'll be reviewing in future issues), where you have automobiles or an opposing team or missiles coming at you, and you've *got* to make fast decisions *right now.*

Fast decisions are usually required, however, in group play with Comp IV,

unless you don't mind being called a slowpoke or an idiot or worse.

Group Play

According to the Comp IV manual, the game can be played by a group in two ways. In Rotation Play, each player makes an entry, announces it to the group, and uses a sheet of the entry pad to keep track of the game's progress. "For large groups of experienced players, the five-digit game is often best." The first person to guess the number that Comp IV holds is the winner.

In Repeat Play, advantage is taken of a circuit that allows Comp IV to hold a number for repeated games, so that each player gets a chance for a complete game on that number. Each player guesses the correct number in secret, then presses E instead of R. This will keep the same number in Comp IV, instead of generating a new one. The player solving the secret number in the least number of steps is the winner. (A counter to indicate how many times the E key is pressed would be a big help, by showing exactly how many steps were required to find the secret number, but this would also increase the price of the game.)

Observations

Obviously, Comp IV has to generate a five-digit number each time, because it doesn't know if you're going to play a three-, four-, or five-digit game. So it may come up with 45283, but if you're playing a three-digit game, you'll win with 452.

The three-digit game is quite simple. It can be played without looking at the sequence lights, although such a game takes a little longer without the hints provided by these lights. After a three-digit game, some pessimists may think Comp IV is a child's toy. Let them try a four-digit game, and especially a five-digit game. They may well require a game pad, which can shorten the game for those who find they just can't keep all the "sure" and "out" numbers in their heads. Even with a game pad, there will no doubt be people who simply give up in frustration.

There is a way to cheat. The manufacturer probably provided this "easy way out" to keep some frustrated player from taking a sledge hammer to his (or somebody else's) Comp IV, or even to keep him from banging his head against the wall. You can build up quite an anger against an 11-ounce box of plastic and metal, if it refuses to divulge its secret number despite all your banging on the keys, and will only blink its little lights.

The manufacturer doesn't call it cheating, but instead politely refers to it as "one strategy that can be used to discover whether certain digits are held." You enter a single digit as many times as there are digits in the game. If you're playing a three-digit game, and want to find out for sure if there's a 7 in the secret number, enter 777, and if 7 was included, you'll get a 3 light in number, and a 1 light in sequence. Press all the other numbers three times, and you'll find the other two. Now all you have to do is figure out the sequence.

The game is so fascinating that you'll soon decide to get an AC adapter, rather than use up a lot of 9-volt batteries.

The random-number generator provides a completely new number each time; there is no stored set of numbers in the IC. The number of possible five-digit combinations of ten digits, with no repeated digits, is 10x9x8x7x6, or 30,240. The possible combination of four digits is 10x9x8x7, or 5,040.

Comparison with Computer Games

In the BAGELS game (*101 BASIC Computer Games)* essentially the same information is provided by the host computer, which indicates with words instead of lights the same information (PICO: one digit is correct, but in the wrong place; FERMI: one digit is in the correct place; BAGLES: no digit is correct).

BULCOW (*101 BASIC Computer Games)* is a somewhat advanced version of BAGELS, known as Bulls and Cows (more popular in England than in the U.S.). "A BULL is scored for each correct digit in the correct position and COW for each correct digit but out of position."

MASTERMIND (Mar-Apr 1976 *Creative Computing*) was originally "simply a commercial adaptation (using colors rather than numbers) of the game Bulls and Cows." Two people play against each other, using a plastic game board and plastic pegs of various colors. Each player tries to guess the color and position of his opponent's pegs concealed in a "hidden code" portion of the game board. *Creative* provides a computerized version.

MASTERBAGELS (Jan-Feb 1977 *Creative Computing*) combines Bagels, Bulls and Cows, Mastermind, and several other games "into a general deductive logic game." If the player wishes to play Bagels, he sets the initial parameters to N,3,9; if Mastermind is his choice, the setting is N,4,6. "But the real fun is trying new combinations."

History

The first Milton Bradley game came out in 1860, and is still available: "The Game of Life," involving growing up, college, a career, etc. (No relation to the game of Life invented in 1970 by Cambridge mathematician John Conway, and popularized by Martin Gardner in *Scientific American.* The whole game is ruled by three very simple genetic laws that define the survival, death and birth of the cells at each generation).

The 1960's opened the big era of plastic games, featuring hundreds of injection-molded items. In 1977, the Milton Bradley Company introduced their first electronic game, Comp IV. The price, depending on when and where you buy the game, is roughly between \$20 and \$40. ■

One of the newest additions to the *Creative Computing* catalog of products is a board game called "Computer Rage," produced in cooperation with Edumatics Corporation. It differs markedly from other computer games published by *Creative Computing* in that it is strictly a *board* game and does not require any access to a computer or terminal to play the game. For this reason, it has appeal to the total Creative Computing audience from the novice hobbyist, to the educator, to the computer professional.

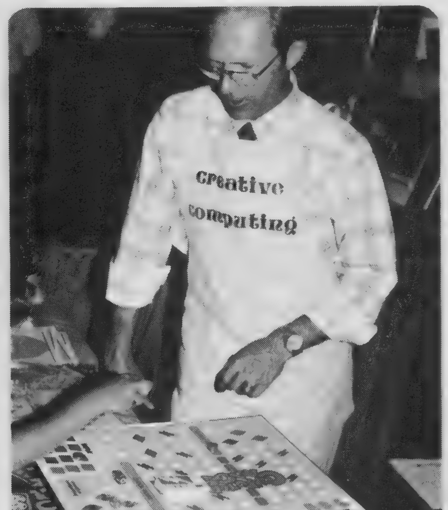
Entertainment

Taken at its most elementary level, Computer Rage can be viewed as another entertaining board game, one that simply has a computer theme behind it. The rules of the game are relatively simple. Absolutely no computer knowledge is required to play the game for enjoyment. It is appropriate for the whole family including both younger children and older adults. While the game professes to be for "Age 10 to Adult," six-year-olds have mastered its fundamentals quickly. It certainly can stand on its own alongside the multitude of board games now found in every toy or department store.

The multi-talented creator reveals why the world's first computer board game intrigues both professors and children.

A Multi-Level Computer-Oriented Board Game

Alan B. Salisbury



Author demonstrating Computer Rage at PC'77 in Atlantic City.

"Computer Rage"

Education

Beyond the entertainment level, Computer Rage provides a significant educational tool for use either in an informal learning environment or in the classroom. Essentially the game is a simulation of a multi-programming or time-sharing computer system providing an opportunity to learn about many fundamental concepts of modern computers. A supplement to the rules, included with the game, explains many of these concepts and how the game relates to the real world.

Computer Rage is played by from two to four players. Each player has three PROGRAMs initially located in the INPUT area. The objective of the game is for each player to move his three programs from INPUT through the processing area, finally arriving at OUTPUT. It is a "throughput" competition, in that the first player to complete the processing and output of his three programs is the winner.

One of the first teaching points of the game derives from the fact that binary dice (containing only 1's and 0's) are used. This provides the opportunity to discuss the internal use of binary numbers within a computer, and the conversion between binary and decimal. Three color-coded dice are utilized with colors corresponding to 4-2-1 weights. A player rolls the dice, converts the binary number to decimal, and then studies his possible moves. The printed rules include a conversion chart, but even the youngest players quickly learn to read the dice directly. (In fact it is the older adults who have the only difficulties here!)

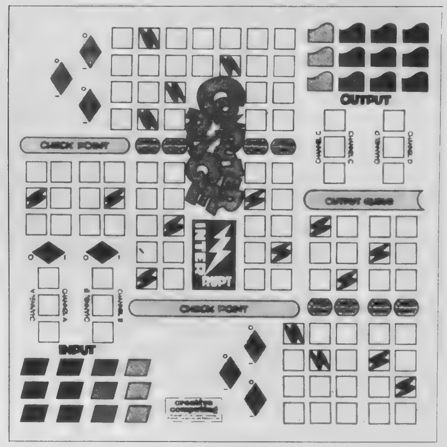
To move out of the INPUT area into the processing area, a PROGRAM must first pass through a CHANNEL. The concept of a channel as a restrictive resource is illustrated here by the rule that only one PROGRAM may occupy or pass through a CHANNEL at any time. Two input CHANNELs are available, so with four players each having three programs, a true resource contention is possible.

Once a PROGRAM has cleared a CHANNEL, it begins its processing journey to OUTPUT. Positions on the board within the processing area include PROCESS symbols, DECISION symbols and CHECKPOINTs interconnected in flowchart fashion. DECISION points alter the flow through the processing area. Each DECISION symbol has two alternative output directions, marked "1" and "0", with the selected path normally determined by the roll of a single binary die. The primary impact of the alternative paths is the number of subsequent PROCESS symbols to be travelled through, with the shortest path being most desirable in terms of throughput. A discussion of the power of decision points in a computer program in altering the subsequent program steps encountered can be based on the use of DECISION symbols in the Computer Rage game.

One of the dangers in a real multiprogramming computer system is interference between programs. This is illustrated by the Computer Rage rule permitting only one PROGRAM to occupy a symbol at any one time, with the exception of the CHECKPOINT symbols (and the OUTPUT QUEUE) distributed over the board. A PROGRAM landing on an occupied symbol must return to the last CHECKPOINT (or back to INPUT if no CHECKPOINTs lie behind the program). This also can be related to the competition for resources.

Throughout the processing area are randomly distributed a number of PROCESS symbols with an overlayed communications symbol indicating the occurrence of an INTERRUPT. A player whose program lands on an INTERRUPT symbol draws a card from the INTERRUPT deck and follows the directions on the card. This provides a vivid demonstration of the use of interrupts to temporarily suspend processing and alter the normal actions of the computer. INTERRUPT actions in the Computer Rage game are about evenly distributed between "good" and "bad" results. Examples include "Efficient Program" cards which advance a PROGRAM, "Program Error" cards which return a PROGRAM back to the last CHECKPOINT, and even a small number of catastrophic failure cards (system "crashes") which can return all programs in process to the last CHECKPOINT. Some INTERRUPT cards can be saved for a "Free Choice Decision" in lieu of die roll, or an "Interrupt Override." Virtually all of the INTERRUPT actions can be used for discussions to make a number of teaching points.

Two output CHANNELs lead to the OUTPUT area and have the same restrictions as the input CHANNELs. An OUTPUT QUEUE (similar in function to a CHECKPOINT) provides a waiting area for programs waiting to



Board layout for Computer Rage, taken from the rules on the bottom of the game box (the game board itself is in full color).

Three *Creative Computing* employees take time off from work to play Computer Rage.

gain access to a CHANNEL. An added restriction is made that a player must roll a number that will exactly allow his program to terminate on one of his unoccupied OUTPUT symbols, or else his PROGRAM must wait in the CHANNEL or OUTPUT QUEUE. (This restriction can be eliminated, if desired, to reduce the impact of the OUTPUT QUEUE on the outcome of the game).

Some of the functions of an operating system can also be illustrated through the play of Computer Rage. The orderly rotation of turns between players can be viewed as a form of "round robin" scheduling. Allocation of "processing time" is actually done by the roll of the binary dice which randomly produce numbers between 0 and 7. (A roll of 0 effectively causes a loss of turn, but life is like that! A suggested change to the rules would make a roll of 0 produce an automatic INTERRUPT, applicable to the roller's program furthest advanced, but not in OUTPUT.) Once the dice have been rolled, the player participates in the scheduling process by deciding which of his three PROGRAMs he will move. This adds some degree of skill to the play of the game, since some strategies can be more successful than others. The INTERRUPT process can also be related to the operating-system functions with some cards providing "Scheduling Change" actions, resulting in either lost or extra turns.

More Entertainment

For those who are already knowledgeable of all the concepts embodied in the Computer Rage game, it can be played at what might be termed the "sophisticated entertainment" level. Play by a wide variety of computer professionals has confirmed that there is a sufficient number of parallels between the game and the real world to make both equally frustrating — hence one reason for the "Rage" in the name of the game. The inside jokes generated during the play by computer freaks are legion.

Computer Rage offers something for almost everyone, whether it be entertainment, education or both. As a pure board game it has the advantage of not requiring access to an actual computer. It is probably only a matter of time, however, until computer implementations begin to appear for those who have computers. (This would certainly be an interesting project, and *Creative Computing* would welcome listings of BASIC COMPUTER RAGE programs.) In the meantime, the whole family or class can get involved with the new Computer Rage board game, and not have to worry about energy consumption or power failures! ■

Why I Did It

The compiler of the S-100 Computer Kit Reference List explains....

Robert Elliott Purser

In 1967, I picked up a free copy of the PDP-8 manual and have been hooked on computers ever since.

Having studied civil engineering, enology, and accounting, I went on to become a computer systems analyst, a mail-order entrepreneur on home knitting, a weekend manager of a computer store, and when times get rough—a janitor.

Recently I left my programming job with the state government to do "something useful for a change." Now I am working on all the projects I dreamed up in the last three years but never had the time to do. Besides keeping up the S-100 Computer Kit Reference List (next issue Jan. 78), I am starting another reference list. This one will catalog all software available on cassettes for Sol-20, PET, Radio Shack, and Apple II. People have asked my why I compile these lists and all I can tell them is, "It needs to be done and no one else has done it."

Since I am one of the nation's leading experts on programming designs for home knitting looms, one of the projects I am now working on is designing knits with hobby computers. In cooperation with a nationwide knitting-loom magazine, I am creating garment patterns on my computer and having them tested by knitters all across the country. My computer is an Altair 8800A, with 8K Seals, Morrow Intelligent Cassette, PolyMorphics video with keyboard, Southwest Technical printer, and most importantly, Li-Chen Wang's Palo Alto Tiny BASIC. Once the programs work they will be converted to the Sol-20, PET, and Radio Shack. Knitters will be able to create garment patterns from their own measurements and from the weight of their own yarn. (Can't you just see Granny in a rocking chair next to the fireplace, knitting patterns displayed on her computer?)

At the same time I am writing programs to emulate the computer-on-cardboard games (such as Code Name: Sector from Parker Brothers, and Electronic Battleship from Milton Bradley). Maybe with a little prodding, these game giants will realize that a $5 cassette is cheaper than a $30 game, as well as more fun.

Then my final project, if no one else has done it, is to develop a computer program by which people set type on their own home computers. A person could key in typesetting instructions on a Sol-20 (it has upper and lower case), send the cassette to a phototypesetting plant, and receive back by mail the text, ready for paste-up.

Right now, for my S-100 Reference List I use a Xerox 800 word-processing unit with a Qume proportional spacing print wheel. By using a word-processing unit I am able to cut cost and time to one-tenth that of typesetting. If I were able to typeset on my own computer, the cost would drop even further.

I am looking for sponsors for my projects. When my savings are depleted, I will be looking for employment. I want to be involved in the more interesting areas of computing, such as developing application programs for home computers or ADAM. (ADAM is the large micro computer system from Logical Machine Corp. The ADAM language is to COBOL as BASIC is to Fortran.)

Or perhaps I will start a mass-market software company. If PET or Radio Shack computers reach their expected sales of 10,000 units a month, a mass-marketing software company can sell a lot of $5 cassettes, since every computer owner will probably buy at least three program cassettes for his computer. At the same time, substantial royalties will be paid to freelance programmers. The possibilities are ENDLESS!

S-100

Bus Compatible

Computer Kits

The S-100 Bus lists computer kits which plug directly into the S-100 bus. Only kit prices are listed unless the board is only available assembled. All products and prices are based on manufacturers' and dealers' advertisements, catalogs, etc. I try to be accurate but I sometimes make mistakes. Please let me know if I do.

Manufacturers: Please send me your catalogs. How else will I know about your S-100 products?

COMPUTER SYSTEMS

Byte Shop Byt-8	349.00
Computer Power & Light COMPAL-80 (assembled)	2,300.00
Cromemco Z-1 (assembled)	2,495.00
Cromemco Z-2K	595.00
Electronic Control Technology ECT-100-8080	320.00
Electronic Control Technology ECT-100-Z80	420.00
Equinox 100	699.00
Forethought Products KIMSI, connector and KIM (6502)	370.00
IMSAI 8080 Computer (chassis, power, & CPU)	699.00
IMSAI PKG-1	4,444.00
IMSAI PKG-2	9,013.00
MITS Altair 8800B	875.00
Morrow's Micro Stuff Sigma 100	250.00
PolyMorphic Systems POLY-88 System 0	525.00
PolyMorphic Systems POLY-88 System 2	735.00
PolyMorphic Systems POLY-88 System 6	1,575.00
PolyMorphic Disk System (1 disk)	3,250.00
Processor Technology SOL-PC Single Board	475.00
Processor Technology SOL-10 Terminal Computer	795.00
Processor Technology SOL-20 Terminal Computer	995.00
Processor Technology System I	1,649.00
Processor Technology System II	1,883.00
Processor Technology System III	4,237.00
Quay AI Z-80 CPU, SIO, PIO, ROM, Programmer Board	450.00
Technical Design Labs XITAN Alpha 1	769.00
Technical Design Labs XITAN Alpha 2	1,369.00
Vector Graphics Vector 1	699.00
Vector Graphics Vector 1 without PROM/RAM	519.00
Vector Graphics Vector 1 without CPU	499.00
Vector Graphics Vector 1 without CPU, PROM/RAM	349.00
Western Data Systems DATA HANDLER (uses MOS 6502)	179.95
Western Data Systems DATA HANDLER (barebones)	79.95

SECOND OR REPLACEMENT CPU BOARD

Affordable Computer Products AZPU (uses Z-80)	249.00
Alpha Micro Systems AM-100 (16 bit)	1,495.00
CGRS 6502	?
Cromemco ZPU (uses Z-80/4 microprocessor)	295.00
IMSAI MPU-A (requires additional boards)	190.00
MRS AM6800 CK (uses 6800 MPU)	110.00
MRS AM6800 (without the 6800 MPU chip)	78.00
MRS AM6800 PC Board	30.00
R.H.S. Marketing Piggy-Back Z80-80 (assembled)	159.95
SD Sales Z-80 CPU	149.00
Technical Design Labs Z-80 (uses Z-80)	269.00

Edition 7, June 1977

I am seeking financial support for my public service projects. If you find this reference list useful, tell a patron of the arts and sciences about my work.

Robert Elliott Purser

READ/WRITE MEMORY BOARD

Advanced Microcomputer Products Logos 8K RAM	219.95
Advanced Microcomputer Products 801C 8K RAM	207.95
Advanced Microcomputer Products 32K RAM	1,150.00
Artec 32K Memory Board (8K, 250 nS)	290.00
Artec 32K Memory Board (32K, 250 nS)	1,055.00
Associated Electronics 16K Pseudo-Static	349.95
Base-2 8KS-A	98.00
Base-2 8KS-B (450 nS)	123.00
Base-2 8KS-Z	143.00
BISI CCD Board (64K)	190.00
Crestline Micro Systems (8K, low power, assembled)	179.00
Cromemco 4KZ (4K 4MHz) (Bank selectable)	195.00
Cromemco 16KZ (16K 250 nS access and cycle)	495.00
Cybercom MB6A Blue Board (8K static)	250.00
Cybercom MB7 (16K low power static)	525.00
Data Sync 16K (assembled)	298.00
Duston 8K Memory Board (bare)	29.00
Dutronics 4KLST (4K low power static)	139.00
Dutronics 8KLST (8K low power static)	285.00
E.E. & P.S. 8K (8K static)	295.00
E.E. & P.S. 16K (16K dynamic)	599.00
E.E. & P.S. 32K (32K dynamic)	895.00
Electronic Control Technology 8KM (8K 215 nS)	295.00
Electronic Control Technology 16K RAM (16K static)	555.00
Electronic Control Technology 16K RAM (with only 4K)	169.00
Electronic Control Technology 16K RAM (with only 8K)	295.00
Electronic Control Technology 16K RAM (with only 12K)	425.00
Extensys RM64-32 (32K)	895.00
Extensys RM64-48 (48K)	1,195.00
Extensys RM64-64 (64K)	1,495.00
Franklin Electric 8K Static RAM	225.00
Godbout Econoram (4K static)	99.95
Godbout Econoram II (8K)	163.84
IMSAI RAM 4A-4 (4K without sockets)	139.00
IMSAI RAM 4A-4 (4K with sockets)	159.00
IMSAI 65K (dynamic)	2,599.00
IMSAI 32K (dynamic)	749.00
IMSAI 16K (dynamic)	449.00
Kent-Moore 4K (assembled)	107.00
Microdesign MR8 (EPROM/RAM	124.95
Micromation JUMP START (4K static)	145.00
Midwest Scientific Instruments PROM/RAM Board	95.00
Mikra-D MD-2046-4 (4K static)	205.00
Mikra-D MD-2046-8 (8K static)	345.00
Mikra-D MD-2046-12 (12K static)	485.00
Mikra-D MD-2046-16 (16K static)	625.00
MiniMicroMart C-80-4K-100 (4K blank board)	39.95
MiniMicroMart C-80-4K-200 (4K blank board plus)	49.95
MiniMicroMart C-80-4K-300S (4K 2102)	79.95
MiniMicroMart C-80-4K-300LP (4K 91L02A)	99.95
MiniMicroMart C80-4K-350LP (4K 91L02C)	129.95
MiniMicroMart C80-16K-300 (16K EMM4200)	479.95
MITS 88-4MCS (4K static)	167.00
MITS 88-16MCS (16K static)	765.00
MITS 88-S4K (4K dynamic)	155.00
Morrow Intelligent Cassette (512 static)	96.00
Mountain Hardware PROROM (256)	164.00
Omni (16K static)	459.00
Omni with paging option (16K static)	468.00
Prime Radix 40K (dynamic)	1,490.00
Prime Radix 48K (dynamic)	1,580.00
Prime Radix 56K (dynamic)	1,670.00
Prime Radix 64K (dynamic)	1,750.00
Processor Technology 4KRA (4K static with sockets)	154.00
Processor Technology 8KRA (8K static with sockets)	295.00
Processor Technology 16KRA (16K static assembled)	529.00
PolyMorphic Systems MEM-8K (8K static)	300.00
R.H.S. Marketing DYNABYTE 16K (dynamic, assembled)	485.00
J-K Electronics DYNA-RAM 16 (16K)	339.00
S. D. Sales Company 4K (4K static)	89.95
Seals Electronics 8KSC-8 (8K static)	269.00
Seals Electronics 8KSC-Z (8K 250 nS)	295.00
Seals Electronics 8KSCLM (less memory chips)	124.00
Seals Electronics 16KSC-16 (16K static)	579.00
Solid State Music MB-4 (4K 91L02A)	129.95
Solid State Music MB-4 (8K 91L02A)	209.00
Solid State Music MB-4 (board only)	30.00
Solid State Music MB-6 (board only)	35.00
Solid State Music MB-6 (8K 91L02APC static)	265.00
Solid State Music MB-7 (16K static)	525.00
Technical Design Labs Z8K (4K 215 nS)	169.00
Technical Design Labs Z8K (8K 215 nS)	295.00
Technical Design Labs Z12K (12K 215 nS)	435.00
Technical Design Labs Z16K (16K 215 nS)	574.00
Technical Design Labs Z Monitor Board with 2K RAM	295.00
Vandenberg 16K RAM (dynamic)	299.00
Vector Graphics 8K RAM	265.00
Vector Graphic Reset and Go PROM/RAM	89.00
Xybek PRAMMER (256 bytes & 1702 PROMs)	189.00

PROM PROGRAMMER BOARD

Cromemco BYTESAVER for 2704 & 2708	145.00
Mountain Hardware PROROM (AMI 6834)	164.00
Quay AI Z-80 with 2708 Programmer	450.00
Szerlip Enterprises The Prom Setter (1702A and 2708)	165.00
Xebek PRAMMER for 1702 (with 1702 & RAM)	209.00

PLUG IN SOFTWARE BOARD

Computer Kits Power-Start	165.00
Cromemco Z80 Monitor Board with PROM Programmer	220.00
Godbout 8080 Software Board	189.95
Microdesign MR8 with MM2K	224.45
Micronics Better Bug Trap (assembled)	180.00
Midwest Scientific Instruments PROM/RAM Monitor	245.00
Mountain Hardware PROROM	164.00
National Multiplex Corp No. 2 SIO with monitor	140.00
Processor Technology ALS-8 (assembled)	425.00
Processor Technology ALS-8 with SIM-1	520.00
Processor Technology ALS-8 with TXT-1	520.00
Technical Design Labs Z System Monitor Board	295.00
Vector Graphics Reset and Go (2 1702A)	129.00
Vector Graphics Reset and Go (3 1702A)	159.00

PROM BOARD

Crea Comp M 100/16 (16K, 2116)	485.00
Crea Comp M 100/16 (with parity)	560.00
Crea Comp M 100/32 (32K, 2116)	885.00
Crea Comp M 100/32 (with parity)	990.00
Cromemco BYTESAVER (8K)	145.00
Cromemco 16KPR-K (16K, Bank selectable)	145.00
DigiComm Byteuser (uses 2708)	65.00
Digiteck PROM CARD (2K assembled without PROMS)	56.95
Electronic Control Technology 2K ROM/2K RAM	120.00
Godbout Econorom (2K)	135.00
Godbout Econorom (4K)	179.95
Godbout Econorom (8K)	269.95
IBEX 16K PROM Board	85.00
IMSAI PROM 4-4 (4K PROM)	399.00
IMSAI PROM 4-512 (1/2K PROM)	165.00
Microdesign MR8 (for 2708)	99.50
Midwest Scientific Instruments PROM/RAM Board	95.00
MiniMicroMart C80-1702-1 (all except PROMS)	49.95
MiniMicroMart C80-2708-2 (all except PROMS)	49.95
MiniMicroMart C80-256 (boot strap board, fuse link)	34.95
MITS PMC (2K)	85.00
Processor Technology 2KRO	65.00
Seals Electronics 4KROM	119.00
Solid State Music MB-3 2K (8 1702As)	105.00
Solid State Music MB-3 4K (16 1702As)	145.00
Solid State Music MB-3 (without PROMs)	65.00
Solid State Music MB-8 (2708)	85.00
Vector Graphic Reset and Go PROM/RAM	89.00
Xybek PRAMMER for 1702 (with a 1702 & RAM)	189.00

MEMORY CONTROL BOARD

IMSAI IMM ROM Control Kit	299.00
IMSAI IMM EROM Control Kit	499.00

PARALLEL INTERFACE BOARD

Advanced Microcomputer Products (3P+S compatible)	125.00
Cromemco D+7AIO (one port with seven analog ports)	145.00
Cromemco TU-ART (2 ports)	195.00
IMSAI PIO 4-1 (one port without cables)	93.00
IMSAI PIO 4-1 & PIOM (two ports without cables)	115.00
IMSAI PIO 4-1 & PIOM (three ports without cables)	137.00
IMSAI PIO 4-4 (four ports without cables)	156.00
IMSAI PIO 6-3 (three ports and bus without cables)	139.00
IMSAI PIO 6-6 (six ports and bus without cables)	169.00
IMSAI MOI (two ports & serial & tape interface)	195.00
MicroLogic M712 (one port)	69.95
MiniMicroMart C80-P I/O (two ports)	49.95
MiniMicroMart C80-P I/O with cables C80-P I/O-540	57.45
MITS 88-4PIO (one port)	105.00
MITS 88-4PIO+PP (two ports)	148.00
MITS 88-4PIO+2PP (three ports)	191.00
MITS 88-4PIO+3PP (four ports)	234.00
Morrow Intellegent Cassette with one port	102.00
PolyMorphic VTI/32 (one input port with video)	185.00
PolyMorphic VTI/64 (one input port with video)	210.00
Processor Technology 3P+S (with sockets)	149.00
Solid State Music I/O-1 (one port)	42.00
Solid State Music I/O-1 (PC board only)	25.00
Solid State Music I/O-2 (two ports)	47.50
Solid State Music I/O-2 (PC board only)	25.00
Technical Design Labs Z Monitor Board (one port)	295.00
WIZARD PSIOB (3P+S compatible)	125.00

SERIAL INTERFACE BOARD

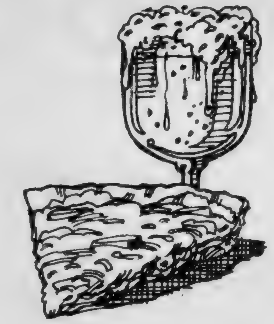
Advanced Microcomputer Products (3P+S compatible)	125.00
Cromemco TU-ART (2 ports)	195.00
IMSAI SIO 2-1 (one port, without cables)	125.00
IMSAI SIO 2-2 (two ports, without cables)	156.00
IMSAI SIO (serial, parallel, & tape interface)	195.00
Morrow Intellegent Cassette with one port	108.00
MiniMicroMart C80-SI/O-300 (TTL)	44.95
MITS 88-2SIO (one port)	150.00
MITS 88-2SIO+SP (two ports)	188.00
MITS 88 SIOB	124.00
National Multiplex Corp No. 2 SIO with ROM	140.00
Processor Technology 3P+S (with sockets)	149.00
Solid State Music I/O-2 (two ports)	47.50
Solid State Music I/O-2 (PC board only)	25.00
Technical Design Labs Z Monitor Board (two ports)	295.00
WIZARD PSIOB (3P+S compatible)	125.00

ANALOG INTERFACE BOARD

Cromemco D+7AIO (7 analog inputs & 7 outputs)	145.00
Micro Data ADC/DAC	250.00
MITS 88-ADC (assembled only)	524.00
MITS 88-Mux (assembled only)	319.00
MITS AD/DA (assembled)	235.00
PolyMorphic Systems ADA/1 (1 analog output)	145.00
PolyMorphic Systems ADA/2 (2 analog outputs)	195.00

MODEM BOARD

International Data Systems 88-MODEM	199.00
Hayes 80-103A (assembled)	279.95
Hayes 80-103A (board only)	49.95



AUDIO CASSETTE INTERFACE BOARD

Affordable Computer Products Triple Standard	135.00
DAJEN Cassette Interface	120.00
DAJEN Universal Cassette Interface (Relay Control)	135.00
IMSAI MIO (tape interface, parallel, & serial)	195.00
MiniTerm Associates MERLIN with cassette interface	298.00
MITS 88-ACR	145.00
National Multiplex Corp No. 2 SIO with ROM	140.00
Morrow Intelligent Cassette Interface	96.00
Morrow Intelligent Cassette Interface (3 drives)	102.00
PerCom Data CI-812	89.95
Processor Technology CUTS	87.00
RO-CHE with Tarbell (two ports)	215.00
RO-CHE with Tarbell (four ports)	245.00
Tarbell	120.00

TAPE DRIVE INTERFACE BOARDS

MECA ALPHA-1 System	400.00
Micro Design Model 100 (assembled)	600.00
Micro Design Model 200 (assembled)	875.00
MicroLogic M712 DG PhiDeck	69.95
National M.C. 2 SIO (R) 1 ROM	169.95
National M.C. 2 SIO (R) 2 ROM	189.95
National M.C. 2 SIO (R) with 3M3 (3M drive)	369.90
National M.C. 2 SIO (R) with 3M3 (mini 3M drive)	339.90

FLOPPY DISK INTERFACE BOARD

Alpha Micro Systems AM-200 Controller	695.00
Alpha Micro Systems AM-201 Controller	695.00
CHP Floppy Disk Controller	300.00
Computer Hobbyist Products Controller	300.00
Computer Hobbyist Products (single drive)	850.00
DigiComm 8040 Floppy Disk Controller	265.00
Digital Systems IBM compatible	1,595.00
Digital Systems dual IBM compatible	2,170.00
iCOM Microfloppy Model FD2411 (assembled)	1,095.00
IMSAI FIF	599.00
IMSAI FDC2-1 & FIF	1,694.00
IMSAI FDC2-2 & FIF	2,789.00
INFO 2000 Adapter (without RAM)	120.00
INFO 2000 Adapter (with 4K RAM)	160.00
INFO 2000 Adapter + Per Sci 1070 Controller	860.00
Micromation Universal Disc Controller	229.00
Micromation MACRO DISC System, Model 164K	900.00
Micromation MACRO DISC System, Model 256K	1,100.00
Micropolis 1053 Mod II (630K)	1,795.00
Micropolis 1043 Mod II (315K)	1,095.00
Micropolis 1053 Mod I (286K)	1,545.00
Micropolis 1043 Mod I (143K)	945.00
MITS 88-DCDD (controller & disk)	1,425.00
MITS 88-DISK	1,215.00
North Star Computers MICRO-DISK	699.00
PerCom Data Co.	695.00
Peripheral Vision interface and floppy	750.00
Peripheral Vision IFF-KC interface	245.00
Pertec RD2411	1,095.00
Processor Applications FDC-1016K Controller	395.00
Processor Technology Helios (dual)	1,895.00
Realistic Controls Z//25	1,095.00
Synetic Designs interface and floppys	2,690.00
Tarbell Bare Board Interface	40.00
Tarbell Interface	190.00

HARD DISK INTERFACE BOARD

IMSAI DISK-50	12,500.00
IMSAI DISK-80	14,700.00
IMSAI DISK-200	24,500.00
IMSAI Interface (assembled)	3,900.00

VIDEO INTERFACE BOARD - BLACK & WHITE

Computer Kits INTELLITERM (characters)	395.00
Computer Graphics GDT-1 (graphics and light pen)	185.00
Environmental Interface II (monitor)	245.00
Environmental Interface III (oscilloscope)	495.00
Kent-Moore alpha (assembled)	107.00
Kent-Moore graphic (assembled)	137.00
Micro GRAPHICS "THE DEALER" (graphics and characters)	249.00
MiniMicroMart C80-VBA	149.95
MiniTerm Associates MERLIN (without memory)	269.00
MiniTerm Associates MERLIN (with memory)	303.95
MiniTerm Associates MERLIN Super Dense Graphics	308.00
Polymorphics VTI/64 (graphics and characters)	210.00
Processor Technology VDM-1 (characters)	199.00
Solid State Music 64x16 (graphics and characters)	179.95

VIDEO INTERFACE BOARD - COLOR

Cromemco TV DAZZLER (graphics)	215.00

TV CAMERA INTERFACE BOARD

Cromemco 88-CCC-K	195.00
Cromemco 88-CCC-K with Camera Kit 88-ACC-K	390.00
Environmental Interface I	295.00
Environmental Interface with camera	595.00

HARDWARE MULTIPLY/DIVIDE BOARD

GNAT 8006 Module (5 u-sec. process time)	225.00
GNAT 8006 Module (2.5 u-sec. process time)	275.00
North Star Computers (floating point)	359.00

CALCULATOR INTERFACE BOARD

COMPU/TIME CT 100	195.00
COMPU/TIME C 101	149.00
MiniMicroMart C80-SCI-300	99.95

SPEECH SYNTHESIZER BOARD

Ai Cybernetic Systems Model 1000	325.00
Computalker Speech Synthesizer CT-1	395.00
Logistics Synthesizer (multipurpose)	525.00

SPEECH RECOGNITION BOARD

Heuristic Speechlab	245.00
Phonics SR/8 (assembled)	550.00

JOYSTICK INTERFACE KITS

Cromemco Joystick Kit & D+7AI/O	210.00
Cromemco Dual Joystick Kits & D+7AI/O	275.00

INTERUPT BOARD

Cromemco TU-ART	195.00
El Paso Computer Group (board only)	20.00
IMSAI PIC-8 (with internal clock)	125.00
MITS 88-VI/RTC	138.00

REAL-TIME CLOCK

Comptek CL2400	98.00
COMPU/TIME CT 100	195.00
COMPU/TIME T 102	165.00
International Data Systems SMP-88	96.00
Lincoln Semiconductor Clock and Display Driver	95.00

AC POWER CONTROL

Comptek PC3216 Control Logic Interface	189.00
Comptek PC3216 & PC3202 Power Control Unit	228.50
Comptek PC3216 & 16 PC3202 16 Channel System	821.00
Comptek PC3232 Control Logic Interface	299.00
E.E. & P.S. 115V I/O	249.00
Mullen Relay/Opto Isolator Control Board	117.00

BATTERY BACK-UP BOARD

Seals Electronics BBUC (12 amper hours)	55.00
E.E. & P.S.	55.00

MUSIC SYNTHESIZER BOARD

ALF Quad Cromatic Pitch Generator (1 channel)	111.00
ALF Quad Cromatic Pitch Generator (2 channels)	127.00
ALF Quad Cromatic Pitch Generators (3 channels)	143.00
ALF Quad Cromatic Pitch Generator (4 channels)	159.00
Cybercom SB1 Synthesizer Kit	250.00
Galaxy Systems MG-1	299.00
Logistics Synthesizer (multipurpose)	525.00
SRS Polyphonic Synthesizer SRS-320 (assembled)	175.00
SRS Polyphonic Synthesizer SRS-321 for the SRS-320	175.00

PRINTER INTERFACE BOARD

Peripheral Vision PRT-KC Printer Kit	495.00

FREQUENCY COUNTER BOARD

International Data Systems 88-UFC	149.00

IBM SELECTRIC INTERFACE BOARD

Micromation TYPEAWAY	225.00

PROTOTYPE BOARD

Advanced Microcomputer Products Universal Proto	39.95
Artec GP-100	20.00
Cromemco WWB-2K	35.00
Electronic Control Technology PB-1	22.00
E.E. & P.S. Wire Wrap	39.00
E&L Instruments Breadboarding/Interfacing Station	241.50
Electronic Control Technology PB-1	28.00
Galaxy Systems PB-1	30.00
Homestead Technology HTC-88P (QT sockets)	138.00
Homestead Technology HTC-88PF (foil pattern)	38.00
IMSAI GP-88	39.00
IMSAI BBC-5 & PI06-6 Intellegent Breadboard System	699.00
IMSAI BBC-3 & PI06-3 Intellegent Breadboard System	464.00
MiniMicroMart C-80-WW (wire wrap type)	19.95
MiniMicroMart C-80-DIP (for point to point)	18.95
MiniMicroMart C-80-BUS-WW (wire wrap)	21.95
MiniMicroMart C-80-BUS-WW-125 (with components)	27.45
MiniMicroMart C-80-DIP-BUS (for point to point)	20.95
MiniMicroMart C-80-DIP-BUS-125 (with components)	26.45
MITS 88-PPCB	45.00
MITS 88-WWB	20.00
PolyMorphics Poly I/O	55.00
Processor Technology WWB	40.00
Sargent's Dist. Co.	25.00
Seals Electronics WWC	37.50
Tarbell Electronics	28.00
Vector 8800V	19.95
Vector 8800-A	29.95
Vector 8800-B	89.00

EXTENDED BOARD

Advanced Microcomputer Products Extender	34.95
Artec EXT-100	12.00
Cromemco EXC-2	35.00
E.E. & P.S. Extender W/C	30.00
Galaxy Systems EX-1	25.00
IMSAI EXT	39.00
MiniMicroMart C-80-EXC	24.95
Mullen (with logic probe)	35.00
Processor Technology EXB	35.00
Seals Electronics EXT	29.00
Solid State Music (less connectors)	8.00
Solid State Music (w/w connector)	12.50
Suntronics EXT-1	9.95
Vector 3690-12 (assembled)	25.00

ADAPTER BOARD

MiniMicroMart C80-BA (for MOD 8/C-MOD 80 boards)	19.95
Forethought Products KIMSI (for KIM)	125.00

CARD CAGE AND/OR MOTHERBOARD

Advanced Microcomputer Products 8 slot MB w/connectors	79.95
Byte, Inc. Byt-8	229.00
Computer Data Systems Versatile CRT (assembled)	699.95
Electronic Control Technology ECT-100	100.00
Electronic Control Technology MB-20	60.00
Godbout Motherboard (10 slot)	85.00
Godbout Motherboard (18 slot)	118.00
Integrand Research Corp. 800	200.00
Integrand Research Corp. 800A	275.00
MiniMicro Mart Expander (4 slots)	10.95
MiniMicroMart Expander (9 slots)	17.95
Morrow MotherBoard	76.00
Objective Design Crate Book (plans only)	19.95
PolyMorphic P88 Chassis	235.00
TEI Model MCS-112	316.00
T&H Engineering Low Cost Buss	149.00
Vector 18 Slot Motherboard	49.00

TERMINATION BOARD

Godbout	25.00

Affordable Computer Products
Byte Shop #2
3400 El Camino Real
Santa Clara, CA 95051
(408) 249-4221

Advanced Microcomputer Products
P.O. Box 17329
Irvine, CA 92713
(714) 558-8813

Ai Cybernetic Systems
P.O. Box 4691
University Park, NM 88003

ALF Products, Inc.
128 S. Taft
Lakewood, CO 80228

Alpha Micro Systems
17875 N. SkyPark North
Irvine, CA 92714
(714) 957-1404

Altair (see MITS)

Artec Electronics, Inc.
605 Old Country Road
San Carlos, CA 94070
(415) 592-2740

Associated Electronics
12444 Lambert Circle
Garden Grove, CA 92641
(714) 539-0735

Base-2, Inc.
P.O. Box 9941
Marina del Rey, CA 90291

BISI
J.R. Broom
Vancouver, B.C.

Byte Shop
1450 Koll Circle, #105
San Jose, CA 95112

CGRS Microtech, Inc.
Unknown

CHP, Inc.
P.O. Box 18113
San Jose, CA 95158

Comptek
P.O. Box 516
La Canada, CA 91011
(213) 790-7957

Computalker Consultants
P.O. Box 1951
Santa Monica, CA 90406

Computer Data Systems
English Village, Atram 3
Newark, DE 19711

Computer Kits Inc.
1044 University Avenue
Berkeley, CA 94710
(415) 845-5300

Computer Graphics Associates
56 Sicker Road
Latham, NY 12110

Computer Hobbyist Products, Inc.
P.O. Box 18113
San Jose, CA 95158
(408) 629-9108

COMPU/TIME
P.O. Box 417
Huntington Beach, CA 92648
(714) 638-2094

Computer Power & Light
12321 Ventura Blvd.
Studio City, CA 91604
(213) 760-0405

Crea Comp System, Inc.
Suite 305
4175 Veterans Highways
Ronkonkoma, NY 11779
(516) 585-1606

Crestline Micro Systems
P.O. Box 3313
Riverside, cA 92519

Cromemco
2432 Charleston Road
Mountain View, CA 94043
(415) 964-7400

Cybercom
2102A Walsh Avenue
Santa Clara, CA 95050
(408) 246-2707

DAJEN
David C. Jenkins
7214 Springleaf Court
Citrus Heights, CA 95610
(916) 723-1050

Data Sync
201 W. Mill
Santa Maria, CA 93454
(805) 963-8678

DigiComm
6205 Rose Court
Roseville, CA 95678

Digital Systems
1154 Dunsmuir Place
Livermore, CA
(415) 413-4078

Digiteck
P.O. Box 6838
Grosse Point, Michigan 48236

Duston, Forrest
885 Aster Avenue
Palatine, IL 60067

Dutronics
P.O. Box 9160
Stockton, CA 94608

E & L Instruments, Inc.
61 First Street
Derby, Conn. 06418
(203) 735-8774

E.E. & P.S.
Electronic Eng. & Production Service
Route #2
Louisville, Tennessee
(615) 984-9640

Electronic Control Technology
P.O. Box 6
Union City, NJ 07083

El Paso Computer Group
9716 Saigon Drive
El Paso, TX 79925

Environmental Interfaces
3207 Meadowbrook Blvd.
Cleveland, Ohio 44118
(216) 371-8482

Equinox Division
Parasitic Engineering
P.O. Box 6314
Albany, CA 94706
(800) 648-5311

Extensys Corp.
592 Weddell Drive, S-3
Sunnyvale, CA 94086
(408) 734-1525

Forethought Products
P.O. Box 386-A
Coburg, Oregon 97401

Franklin Electric Co.
733 Lakefield Road
Westlake Village, CA 91361
(805) 497-7755

Galaxy Systems
P.O. Box 2475
Woodland Hills, CA 91364
(213) 888-7233

GNAT Computers
8869 Balboa, Unit C
San Diego, CA 12123

Godbout Electronics
Box 2355
Oakland Airport, CA 94614

Hayes
P.O. Box 9884
Atlanta, GA 30319
(404) 231-0574

Heuristic, Inc.
900 N. San Antonio Road
Suite C-1
Los Altos, CA 94022

Homestead Technologies Corp.
891 Briarcliff Road N.E.
Suite B-11
Atlanta, GA 30306

iCOM Division
6741 Variel Avenue
Canoga Park, CA 91303
(213) 348-1391

IBEX
1010 Morse Avenue, #5
Sunnyvale, CA 94086
739-3770

I M S Associates, Inc.
14860 Wicks Blvd.
San Leandro, CA 94577
(415) 483-2093

INFO 2000
P.O. Box 316
Culver City, CA 90230

THE FACTORY

Integrand Research Corp.
8474 Avenue 296
Visalia, CA 93277
(209) 733-9288

International Data Systems, Inc.
400 North Washington Street,
Suite 200
Falls Church, VA 22046
(703) 536-7373

Kent-Moore Instrument Co.
P.O. Box 507
Industrial Avenue
Pioneer, Ohio 43554
(419) 737-2352

Lewis and Associates
68 Post Street, Suite 506
San Francisco, CA 94104
(415) 391-1498

Lincoln Semiconductor
P.O. Box 68
Milpitas, CA 95035
(408) 734-8020

Logistics
Box 9970
Marina Del Ray, CA 90291

North Star Computers
2465 Fourth Street
Berkeley, CA 94710

MECA
7344 Wamego Trail
Yucca Valley, CA 92284
(714) 365-7686

Micro Data
3199 Trinity Place
San Jose, CA 95124

Microdesign
8187 Havasu Circle
Buena Park, CA 90621
(714) 523-8080

Micro Designs, Inc.
499 Embarcadero
Oakland, CA 94606
(415) 465-1861

MicroGRAPHICS
P.O. Box 2189, Station A
Champaign, IL 61820

MicroLogic
P.O. Box 55484
Indianapolis, IN 46220

Micromation
524 Union Street
San Francisco, CA 94133
(415) 398-0289

Micronics, Inc.
P.O. Box 3514
Greenville, NC 27834
(919) 752-7813

Micropolis Corp.
9017 Reseda Blvd.
Northridge, CA 91324
(213) 349-2328

Midwest Scientific Instruments
220 West Cedar
Olathe, Kansas 66061

MIKRA-D, Inc.
P.O. Box 403
Hollister, Mass. 01746
(617) 881-3111

Mini Micro Mart
1618 James Street
Syracuse, NY 13203
(315) 422-4467

MiniTerm Associates
Box 268
Bedford, Mass. 01730

MITS (Altair)
2450 Alamo S.E.
Albuquerque, NM 87106
(505) 243-7821

Morrow's Micro-Stuff
Box 6194
Albany, CA 94706

MRS
P.O. Box 1220
Hawthorne, CA 90250

Mullen Computer Boards
Box 6214
Hayward, CA 94545

Mountain Hardware
Box 1133
Ben Lomond, CA 95005
(408) 336-2495

National Multiplex Corp.
3474 Rand Avenue, Box 288
South Plainfield, NJ 07080

Objective Design, Inc.
P.O. Box 20325
Tallahassee, FL 32304

Omni Systems, Inc.
P.O. Box 7536 Univ. Station
Provo, Utah 84602

PerCom Data Company
4021 Windsor
Garland, TX 75042
(214) 276-1968

Peripheral Vision
P.O. Box 6267
Denver, Colorado 80206
(303) 733-1678

Phonics, Inc.
P.O. Box 62275
Sunnyvale, CA 94086
(408) 735-7622

Polymorphic Systems
737 S. Kellogg
Goleta, CA 94608
(805) 967-2351

Prime Radix Inc.
P.O. Box 11245
Denver, Colorado 80211
(303) 433-5630

Processor Applications, Ltd.
2801 East Valley View Avenue
West Covina, CA 91792
(213) 965-8865

Processor Technology
6200-L Hollis Street
Emeryville, CA 94608
(415) 652-8080

Quay Corporation
P.O. Box 386
Freehold, NJ 07728
(201) 681-8700

Realistic Controls Corporation
3530 Warrensville Center Road
Cleveland, Ohio 44122
(216) 751-3158

R.H.S. Marketing
2233 El Camono Real
Palo Alto, CA 94306
(415) 321-6639

RO-CHE Systems
7101 Mammoth Avenue
Van Nuys, CA 91405

S. D. Sales
P.O. Box 28810
Dallas, Texas 75228

Sargent's Dist. Co.
4209 Knoxville
Lakewood, CA 90713

Scientific Research Instruments
P.O. Drawer C
Marcy, NJ 13403

Seals Electronics
Box 11651
Knoxville, TN 37919
(615) 693-8655

Smoke Signal Broadcasting
P.O. Box 2017
Hollywood, CA 90028

Solid State Music
M I K O S
419 Portofino Drive
San Carlos, CA 94070
(408) 246-2707

Stillman Research Systems (SRS)
P.O. Box 14036
Phoenix, AZ 85063

Suntronics Company
360 Merrimack Street
Lawrence, MA 01843
(617) 688-0751

Synetic Designs Company
P.O. Box 2627
Pomona, CA 91766
(714) 629-1974

Szerlip Enterprises
1414 W. 259th Street
Harbor City, CA 90710

TEI Inc.
7231 Fondren Road
Houston, Texas 77036
(713) 774-9526

T & H Engineering
P.O. Box 352
Cardiff, CA 92007
(714) 753-8568

Tarbell Electronics
20620 South Leapwood Avenue
Suite P
Carson, California 90746
(213) 538-4251

Technical Design Labs Inc.
342 Columbus Avenue
Trenton, NJ 08629
(609) 921-0321

Vandenberg Data Products
P.O. Box 2507
Santa Maria, CA 93454
(805) 937-7951

Vector Electronics Company, Inc.
12460 Gladstone Avenue
Sylmar, CA 91342
(213) 365-9661

Vector Graphic Inc.
717 Lakefield Road, Suite F
Westlake Village, CA 91361
(805) 497-0733

Western Data Systems
3650 Charles Street, No. Z
Santa Clara, CA 95050

WIZARD Engineering
8205 Ronson Road, Suite C
San Diego, CA 92111

Xybek
P.O. Box 4925
Stanford, CA 94305
(408) 296-8188

ROBERT ELLIOTT PURSER POST BOX 466, EL DORADO, CA 95623

TDL Xitan

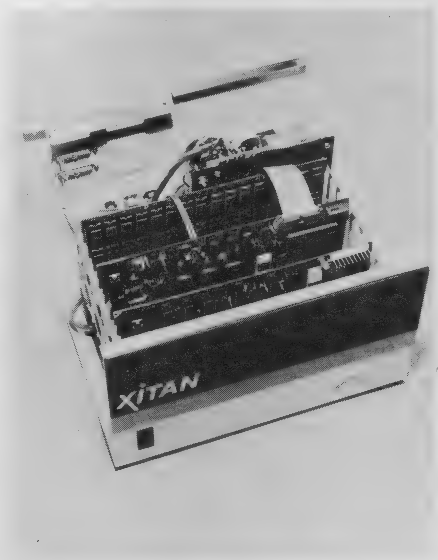
Steve North

Dozens of microcomputers are presently on the market. Many of these are well-designed hunks of hardware, but all too often that's where the manufacturer leaves off with his support of the system. You're left with a nice-looking box and no software to fill it with. But TDL's Xitan Systems come with an impressive array of software which makes it a product of special interest to those who want to spend time writing and using software and not modifying and playing games with hardware.

A Xitan is an S-100 bus computer, built around TDL's ZPU board, which is a CPU board based on the Zilog Z-80 microprocessor. The Z-80 is recognized as the most powerful microprocessor available today and is software-compatible with the Intel 8080, so it has a large body of support software. (It is not entirely accurate to say that the Z-80 is *completely* software compatible with the 8080, since MITS BASIC *won't* run on the Z-80, but just about everything else will. The reason for the problem is that the Z-80 overlaps two flags and assumes that a bit means one thing when a logical operation is done; another when an arithmetic operation is done. MITS BASIC does something tricky with the flags. A patch can be written to get around this if necessary.) The Z-80 has a number of improvements over the 8080 instruction set, including: relative jumps; a test of any bit in any register or memory location; I/O to or from any register; block moves, searches, and I/O; an indirect register, and some instructions they "forgot" when they made the 8080 (such as LDED, similar to LHLD, to load the D and E registers direct). The Z-80 has a duplicate set of registers, called the "prime" registers, which are used to speed interrupt handling, as well as expanded interrupt features. The Z-80 is faster than the 8080 even running at the same clock speed. The ZPU board has two clocks on it, either of which may be used. One is a crystal-controlled 2-MHz clock; the other may be adjusted to any rate from below 1 Hz to above 4 MHz. The ZPU, like the rest of TDL's boards, may be used in any S-100 bus system (such as an IMSAI or Altair). If you plan to do much assembly-language programming, then the Z-80 is a clear choice over the 8080. If you don't care what's inside your machine, you should appreciate that this kind of design philosophy shows up in other places you *will* notice.

System Monitor Board

Another central element of the Xitan is called the System Monitor Board. The SMB is half a computer (almost) on one board. It holds 2K of ROM programmed with a hex monitor called ZAPPLE, and 2K of RAM for scratchpad and extensions to ZAPPLE. Having a monitor on ROM eliminates the need for a front panel and the drudgery of toggling in a bootstrap every time the computer is turned on. When the Xitan is turned on, the processor immediately starts executing ZAPPLE and you're off! The System Monitor Board also has two serial I/O ports (one for a Teletype, one for a CRT) and a parallel port (which might be used with a keyboard or optical papertape reader). In addition the SMB has a cassette interface that uses TDL's phase-encoded asynchronous 1200-baud standard. While this unfortunately means that we have yet another cassette standard to contend with, the interface is almost free (since it was implemented with just 3 ICs). Besides, there is no such thing as a cassette standard, although Tarbell probably comes closest. The SMB cassette interface must be used with a medium-quality (over $50?) cassette unit.



TDL's Xitan Alpha 2 has three boards: ZPU, SMB, and Z16 memory

Z16 Memory Board

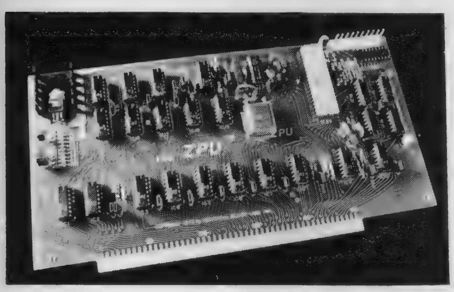
The Xitan Alpha 2 also includes a Z16 memory board. The Alpha 1 doesn't, but is otherwise the same computer. This board holds 16K of dynamic RAM and has an invisible refresh feature so the refresh procedure that dynamic RAMS require won't interfere with other activities of the system. 16K of memory should be adequate for many people, so a TDL Xitan Alpha 2 is a complete system, ready to connect to a terminal, with a CPU, memory, I/O for terminals and a cassette, and a monitor on ROM.

The Xitan comes with a pre-assembled power supply which should be adequate for the eight-slot card cage. The Xitan uses the S-100 bus which means that you can plug all kinds of interesting boards made by other manufacturers into your Xitan (but not too many of them). Eight slots doesn't seem like a lot, though it is difficult to say what the optimum number actually is. Twenty-two (in the IMSAI) seems almost like overkill, whereas five (Processor Technology Sol-20) or eight certainly isn't too many. And the Xitan is down to six slots after you add the ZPU and SMB. The Xitan is in a smaller cabinet than many other systems which makes it easier to move around. Also, only having eight slots cuts down on the cost of the power supply and the case. A RESET switch (which brings you back to ZAPPLE) and a power-on indicator LED are mounted on the front of the cabinet. The on/off switch is on the back of the cabinet (to eliminate the possibility of accidentally switching off the system.)

ZAPPLE Monitor

We previously mentioned a monitor called ZAPPLE which is programmed into the ROM on the System Monitor Board. ZAPPLE includes various useful monitor-type functions. Commands are entered by typing a single character (such as G for GO, or F for FILL) and hexadecimal notation is

used. ZAPPLE has 23 commands for loading, dumping, and filling memory from the keyboard; reading and punching hex and binary files; searching for byte strings in memory, executing user programs, testing memory, etc. The three commands not used by ZAPPLE (I, O, and K) may be used to execute your own user-written commands (I and O obviously suggest some type of I/O operation). ZAPPLE also makes user programs I/O-independent by handling all I/O. To input or output data (a byte at a time) the user's program merely calls the appropriate routine in ZAPPLE. Four logical devices are handled by ZAPPLE: the console, reader, punch, and list device. These may be assigned to various actual devices such as a Teletype, CRT, high-speed papertape reader/punch, cassette, lineprinter, or user-written I/O routine. ZAPPLE permits you to change I/O devices with the "A" command. For example, AC=C assigns the CRT as the console. So a program that does its I/O through ZAPPLE wouldn't actually be concerned with whether output was being printed on a Teletype or



The ZPU card features the Z-80 microprocessor from Zilog.

lineprinter, or whether data was being read from a papertape or a cassette. A DIP-switch on the SMB is used to tell ZAPPLE what I/O configuration to use at system start-up. What if you don't have a terminal? Maybe you want to use a keyboard and VRAM (memory-mapped video display) with the Xitan. In that case, you'd just set the DIP-switch for the BATCH mode. All console input is read from the reader device (which can be selected to be the cassette) and all output goes to the list device (which is non-existent at the moment). Since VRAM devices need to be driven with a user-written I/O routine, you'd merely load a cassette which contains: a command to read a file from the cassette, a file with a VRAM driver routine on it and a command to assign the console to user-written routine. In short, ZAPPLE is a rather powerful monitor by microcomputer standards.

The Xitan Alpha 2 also includes some other powerful software. Alpha 1 systems don't include these packages (but if you don't have additional memory, you can't use them, so for personal computer applications you would be a lot better off with a Z16 memory board and TDL's software too, if you can afford it).

8K BASIC

TDL's 8K BASIC is similar to MITS 8K BASIC, but has its own refinements and comes free with the Alpha 2. It has features found in most powerful microcomputer BASICs: exponentiation, character strings, a complete function library, logical functions (AND, OR, NOT), user-defined functions, program SAVE and LOAD, and complete error messages (not two-letter abbreviations). TDL 8K BASIC also has RENUMBER, program execution tracing, IF THEN ELSE, a SWITCH command to control assignment of the console device, and EDIT, which MITS 8K doesn't have. TDL also offers a 12K BASIC separately, with even more handy features: PRINT USING, double-precision math routines including transcendental functions, multiple-line user-defined functions, and even a SPEAK command for use with a Computalker! As a matter of fact, it's hard to think of anything 12K BASIC *doesn't* have except for MAT functions, which no microcomputer BASIC has. The only negative comment I have about TDL BASIC is that control-X, which is commonly used for a line-delete symbol, is used in TDL BASIC to return to ZAPPLE. I'm sure that once you get used to using TDL BASIC it really doesn't matter that much, but it is inconvenient at first.

Text Editor

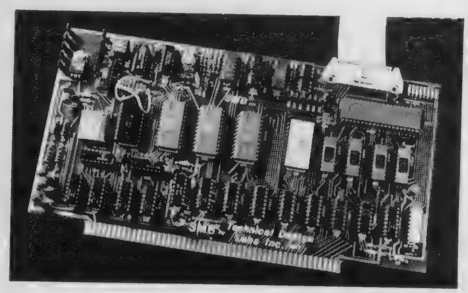
The ZAPPLE Text Editor may be used to edit programs or just general text. Commands are entered as single letters with numeric arguments and may be strung together to form command strings. Operations are done relative to a pointer in the text (for instance, print the next ten lines). The editor permits creation/insertion/replacement/deletion of text and other useful operations. The ZAPPLE Text Output Processor is used, appropriately enough, to control output of text on a hard-copy device. TOP controls pagination, titling, justification, centering, tab settings, indentations, etc.

Macro-Assembler

Finally, Xitan Alpha 2 Systems also include TDL's Z-80 relocating macro-assembler. This is a two-pass assembler with lots of pseudo-ops and the capability to support user-written macros. The pseudo-ops are used to control paging of the assembly listing, linkage between subprograms, macro expansions, global areas, titles, conditional assembly, data assembled in a program, and more. The mnemonics used are patterned on 8080 mnemonics where there is a resemblance, but otherwise conform to Zilog's mnemonics. The assembler can generate relocatable object code which can then be read in at any memory location using a command in ZAPPLE.

In Development

TDL has some other hardware and software products currently in development stages or being readied for release. Among these: a *FORTRAN* IV compiler which should be available by the time you read this; ZTEL, a text-editing *language;* and a Data Base Management System. The Video Display Board is a video RAM device that displays 80 columns and 24 lines, and which doesn't use any of the processor's addressing space. It is programmed through I/O ports, and characters are placed on the screen as fast as they can be written by the computer, so the block-output instruction the Z-80 has may be used. The Video Display Board will also have graphics symbols, video inversion,



The SMB system monitor board uses the Zapple monitor in 2K of ROM.

and blink. Some more memory boards are on the way, as well as a hardware multiply/divide board.

Conclusion

Is the Xitan for you? The microcomputer market has diversified since the Altair came out. A single product is no longer aimed at all microcomputer buyers. If you want to get into microcomputing as cheaply as possibly, there are better routes to take. The Xitan is not really for the same person who would buy a PET 2001 or maybe an APPLE II or a Sol-20. FORTRAN IV and the Data Base Management software really aren't aimed at the personal-computer user at all, which is indicative of the direction TDL is taking with the Xitan (namely, towards greater sophistication and somewhat higher prices). But don't get the wrong idea. If you add up the prices for an IMSAI 8080 with 16K of memory, interfacing for a terminal and a cassette, you'll see that the Xitan is competitively priced and offers a lot more in the software department. ■

AN EVALUATION OF THREE 8080 8K BASICS

Steve North

In choosing a BASIC interpreter or other sophisticated and possibly expensive piece of software for your personal computer, what factor should be kept foremost in mind? A great deal of attention is sometimes paid to timing comparisons. While the speed of an interpreter can be important in some cases, most personal-computer applications tend to be I/O-bound, which makes the internal speed of an interpreter of less significance. Just because one interpreter in 25% faster than another doesn't make it better. However, a well-written interpreter is probably faster than a poorly-written one. The time (or speed) that's really important in this evaluation is *yours*, not the computer's. If it takes you five hours more to code a particular program in one language than in another, thirty seconds hardly make a difference. What most people want is a language that permits them to do what they want with a maximum of flexibility and a minimum of effort (of course, this is only true if you program to do things and not program just to program). Factors such as memory size and availability of assembly-language source code will diminish in importance as memory gets cheaper and people who don't care about assembly language start buying computers. Reliability will become even more important since the average person finds bugs and crashes even more annoying than the computer fanatic does.

This issue we're bringing you an evaluation of three 8080 8K BASICs—MITS 8K BASIC, IMSAI 8K BASIC, and BASIC ETC. In the next issue we'll try to get to some 6800 and maybe Z-80 software, in addition to more 8080 software products (including a micro-APL, a BASIC-like pseudocompiler, a FORTRAN compiler, and a very extended BASIC).

MITS 8K BASIC 4.0

Author: Microsoft.
Size: 6.2K with all functions retained.
Price and Availability: $250, on papertape or MITS cassette. $75 for owners of MITS systems. MITS, Inc., 2450 Alamo S.E., Albuquerque, N.M. 87106
Reliability: No problems noted.
Documentation: The user's documentation for all the MITS BASIC (4K, 8K, and Extended) are lumped together in one manual, forcing the user to separate all the information pertaining to a particular interpreter from useless information on other BASICs. The instructions are above average and include examples. The source code is not available.
Speed: Our original benchmark took 20 minutes to reach 34 and 52 minutes to reach 56. SIDES3 took 43 seconds.

Features:

Commands: CLEAR (clears variables; with an argument, sets amount of string space), CLOAD (loads or checks file on cassette, may also be used as a program statement or command to load arrays from the cassette), CONT (continue after a STOP or control-C), CSAVE (saves a program file or array on cassette), LIST, NEW, NULL (sets number of null characters printed after a CR/LF), RUN.

Statements: DATA, DEF, DIM, END, FOR, GOTO, GOSUB, IF...GOTO, IF...THEN, INPUT, LET, NEXT, ON...GOSUB, ON...GOTO, OUT (outputs a byte to an I/O port), POKE (stores a byte in an absolute memory location), PRINT, READ, REM, RESTORE, RETURN, STOP, WAIT (wait for a specific value at an I/O port).

Variables: Variable names may be any number of characters in length, but only the first two characters are significant. Thus, MITS 8K BASIC doesn't know the difference between the variables DOG and DOUGHNUT, because they both begin with DO. Range is ±2.9387E-38 to ±1.70141E38.

Functions: +,-,*,/,↑, ABS, ASC (returns ASCII value of character string), ATN, CHR$ (converts from ASCII to character string), COS, EXP, FRE (number of free bytes in the system), INP (inputs a byte from an I/O port), INT, LEFT$ (takes left part of a character string), LEN, LOG, MID$ (middle part of a character string), OCT$ (converts a decimal number to a character string of octal digits), RND, POS (position of the print head), RIGHT$ (right part of a character string), SGN, SIN, SPACE$ (creates a string of spaces), SQR, STR$ (converts a number to a string of digits), TAB, TAN, USR, VAL (converts string of digits to a number).

User-defined Functions: Yes, with any number of arguments (such as DEF FNA(X, Y, Z)=X+Y*Z). String functions are also permitted (DEF FNZ$ (X$) =X$+"HELLO").

Arrays: Yes, with any number of dimensions.

Machine-Language subroutine interfacing: Yes, with the USR function. USR has one argument which the machine-language subroutine picks up by calling the subroutine whose address is stored in locations 4 and 5. The address called by USR is set up by POKE-ing memory locations 111 and 112 with the desired address. A value can be passed back to BASIC by placing it in two CPU registers and then calling another subroutine in BASIC.

Character Strings: MITS 8K BASIC has very complete character-string-handling features. Strings may be handled as scalars (X$="HELLO") or as arrays (X$(1,2)="HELLO), etc.) Strings may contain up to 256 characters. The plus symbol (+) is used for concatenation.

Formatted Print: None

Editing Functions: None.

External Files: Only to the extent of being able to save and load arrays from the cassette with a program statement.

Error Messages: 18, specified by a two-character error code.

Extra Stuff: Control-0 is used to control the printing of output on the terminal. "?" is equivalent to PRINT. Control-Q and Control-S are used to temporarily freeze and then continue execution of a program or listing. The logical functions AND, OR, and NOT are available. MITS BASIC has a initialization dialog that permits some functions to be dropped to release more memory for programs. Depending on the sense-switch setting at initialization time, MITS BASIC configures itself for any one of a variety of MITS I/O boards.

User Comment: MITS 8K BASIC was the first widely available microcomputer BASIC and it has remained an outstanding product, a standard with which to compare other BASICs. Unfortunately, its high price discourages its use by low-budget computer hobbyists. But if you can afford it, this is the way to go.

IMSAI 8K BASIC 1.4

Author: IMSAI

Price and Availability: $100 for a papertape or Tarbell cassette. IMSAI Manufacturing Corp., 14860 Wicks Blvd., San Leandro, California 94517.

Size: 8.8K

Documentation: The instructions are merely a list of 8K BASIC's statements and functions with a little explanation, and some random comments on using 8K BASIC. Quite often the instructions are insufficient; for instance, the CHANGE verb is explained by "CHANGE string to array, or CHANGE array to string." Since the DEC BASIC-Plus manual is listed as a reference, one is probably expected to obtain a copy to get a better idea of how to use IMSAI 8K BASIC. The manual includes 113 pages of well-commented source code for 8K BASIC in assembly language.

Reliability: IMSAI 8K BASIC flunks the notorious 10 GOSUB 10 test by recursing itself to death! It seems that IMSAI 8K BASIC never checks to see if it runs out of memory for the BASIC program, variables, or if the stack eats its way down through the BASIC program. In fact, it doesn't even have an out-of-memory error message! There are a few other problems—according to the manual, strings over 238 characters can "destroy the interpreter," and active FOR/NEXT loops may not be restarted. The floating-point math routines also need some help (did you know that 2↑200 is -1.38771E-17? Computers *never* lie!)

Speed: AHLDIG: 89 minutes to reach 34, 160 minutes to reach 56. SIDES3: 103 seconds.

Features:

Commands: CON (continues execution after a STOP or control-C), FRE (prints number of free bytes in system), KEY (used to return to normal mode after a TAPE command), LIST, NEW (an optional form of this command deletes the program but not variables), RUN, SAVE (punches papertape of program), TAPE (loads papertape of program without echoing on the terminal), XEQ (runs program without clearing variables first).

Statements: CALL, CHANGE (convert from string to array or vice versa), DATA, DEF, DIM, END, FOR, GOSUB, GOTO, IF...THEN, IF...GOSUB, IF...GOTO, INPUT, INPUT LINE (used to input a character string with embedded quotes or commas). NEXT, ON...GOSUB, ON...GOTO, OUT (outputs a byte to an I/O port), POKE (stores a byte in an absolute memory location), PRINT, RANDOMIZE (used to true pseudorandom numbers), READ, REM, RESTORE, STOP.

Variables: A-Z, A0-Z9. Range is ±2.7105E-20 to ±5.7646E18.

Functions: +, -, *, /, ↑, ABS, ASCII (returns the ASCII value of a character string), ATN, CHR$ (converts from ASCII to a character string), COS, EXP, INP (inputs a byte from an I/O port), INSTR (used to determine the position of one character string within another), INT, LEFT$ (takes left part of a character string), LEN, LN (natural log), LOG (log base ten), MID$ (middle part of a character string), NUM$ (converts a string of digits into a number), PEEK (returns the value of an absolute memory location), PI (3.14159), POS (returns the position of the print head), RIGHT$ (right part of a character string), RND (without an argument works like normal RND; with an argument, returns a value between one and the argument), SGN, SIN, SPACE$ (creates a string filled with spaces), SQR, STRING$ (same as CHR$, but creates multiple-character strings), TAB, TAN, VAL (converts a number to a string of digits).

User-defined functions: Yes, only one argument is allowed.

Arrays: Arrays may be named A-Z and have one or two dimensions.

Machine-language subroutine interfacing: Yes, but there is no direct way to pass values between BASIC and the subroutine (except by PEEK and POKE).

Character Strings: Strings may contain up to 256 characters, but string arrays are not permitted. The plus symbol (+) is used for string concatenation. Note that the manual reports that there is a bug which causes the interpreter to bomb if strings greater than 238 characters are used. IMSAI 8K BASIC has a complete set of string-handling functions:

Formatted Print: None.

Editing Functions: None.

External Files: None.

Error Messages: 10, indicated by a two-character error code. Overflow and underflow errors are non-fatal (execution of the program continues after a message is printed).

Extra Stuff: "?" and "!" may be substituted for PRINT and REM. Print statements may use single or double quotes for delimiting literal strings. Control-0 is used to control the printing of output. The NEW and XEQ commands provide a rudimentary facility for chaining programs together.

IMSAI also has a expanded version of 8K BASIC 1.4, called 9K BASIC 1.4. This version of BASIC includes CSAVE and CLOAD (cassette save and load of BASIC program) for the Tarbell cassette interface on an IMSAI MIO board, an EDIT command (of the form EDIT (line number) (delimiter) (old text) (delimiter) (new text)), and a BAUD command used to change the terminal baud rate on an MIO board.

User Comment: The apparent intent in writing IMSAI 8K BASIC was to make it as close to DEC BASIC – Plus as possible. In fact, 12K BASIC was supposed to be completely compatible with BASIC-Plus. But alas, along the way, IMSAI 8K BASIC got derailed. There are still

some serious problems that need to be ferretted out before IMSAI 8K BASIC can compare with the quality of IMSAI's hardware products. But we do like the idea, and hope that IMSAI will finally clean up this interpreter. Incidentally, PEEK, POKE, CALL, INP, and OUT are not BASIC-Plus syntax—more like MITS BASIC. Obviously, what's great for some computers may need some alteration for micros. One other comment: we remember reading in IMSAI ads that 4K, 8K, and 12K BASIC would be free to all registered IMSAI owners. Now the price is $100. In its present form, MITS 8K BASIC at $250 seems like a better buy.

BASIC ETC.

Authors: John Whipple and Dick Arnold.
Size: 9K
Price and Availability: On papertape, Kansas City or Suding cassette for $25. Binary Systems, Inc., 6345 Central Expressway, Richardson, 75080.
Documentation: The 30-page instruction manual includes examples and is generally satisfactory. Details on customization are included. The source code is not available.
Reliability: BASIC ETC flunked the 10 GOSUB 10 and the 10 RETURN tests. It also inserted its own line in our benchmark program when a variable overflowed. In none of these cases did the interpreter actually blow up, but it did damage the BASIC program.
Speed: AHLDIG: In real mode, 39 minutes to reach 34, and 100 minutes to reach 56 (the end of the test). In integer mode, about 24 minutes to reach 34. SIDES3 could not be run, because there are no trig functions.
Features:
Commands: LIST, LOAD (loads program and variables (optional) from cassette), NEW, NULL, (sets number of null characters printed after a CR/LF), RUN, SAVE (save program and variables (optional) on cassette).
Statements: DATA, DIM, END, FOR, GOSUB (with an optional list of expressions passed to the subroutine), GOTO, IF, INPUT, LET, MEMW (stores a byte in an absolute memory location), NEXT, OUT (outputs a byte to an I/O port), PRINT, PRMT (sets prompt character used in the INPUT statement), READ, REM, RESTORE, RETURN, SD (sets number of significant digits, 6-72, used internally), STOP, S$SPC (sets amount of string space), VAR (used by a subroutine to pick up arguments passed to it by the calling program), ZONE (sets width of numeric print zone).
Variables: A-Z are integer variables (0 to ± 32767). A0-Z9 are real variables (1E-62 to 9.99...E62).
Functions: +,-,*,/, ABS, FLT (converts from integer to real), FRE (returns the number of free bytes in the system), IN (inputs from an I/O port), INT (converts from real to integer), LEN, MEMR (returns the value of an absolute memory location), RND, SQR, USR, VAL (converts a string of digits to a number).
User-defined functions: Yes, but not stardard BASIC. User-defined functions are numbered FN0 through FN7 and may have one argument that's represented in the function definition by a pound sign (#). For instance, DEF FNZ(X)=27*X in normal BASIC is written as FN2=27*# in BASIC ETC. Strange!
Arrays: Multidimensional arrays are supported. However, only real arrays are permitted and they must have real variable names. Thus, DIM Q(10) is illegal but DIM Q7(N) is OK. The subscripts must be integer mode expressions.
Machine-language subroutine interfacing: Provided by the USR function. Arguments are passed both ways with two CPU registers.
Character Strings: Yes, but they are merely alternate representations of real variables, not distinct. Thus A0 and A0$, and B9(1,2) and B9$(1,2) actually represent the same data. This eliminates the need for string-handling functions. The plus symbol (+) is used to concatenate strings. There is a provision for taking substrings in BASIC ETC. Obviously, not standard BASIC.
Formatted Print: The ZONE command can be used to set the number of positions in the numeric PRINT zone. BASIC ETC also has a non-standard form of PRINT-USING with integer, real, exponential, ane ASCII specifications.
Editing Functions: None.
External Files: None.
Error Messages: 27, specified by a three-digit number (100 to 126).
Extra Stuff: None.
User Comment: BASIC ETC is cheap and it does work, but we don't care for some of its non-standard syntax. For instance, there is no THEN in an IF statement! To write an IF... THEN, you use the colon (multiple-statement separator) in place of THEN, as in IF X=1: PRINT "HI", which probably saved someone three or four bytes in writing BASIC ETC. But the really annoying thing is that BASIC ETC distinguishes between integer and real data types, and you're not allowed to say for yourself if you want X5 to be an integer or a real number. True, integer math in BASIC ETC in significantly faster than real math, but this isn't standard BASIC. Mixing of integer-data types with real-data types is not permitted (so PRINT 2*X5 bombs, because BASIC ETC decides that the expression must be an integer since 2 is an integer). Further, there is no actual INT function in BASIC ETC. To get the integer part of a real variable you can always resort to X5=FLT(INT(X5)), which only works for certain values of X5. We could go on, but you get the idea. BASIC ETC is not, in some areas, anything like any other BASIC we've used. BASIC ETC *is* the most inexpensive floating-point BASIC we've tested so far. ■

New Benchmark Program

Geoffrey Chase

We're introducing a new benchmark program, SIDES3, written by Geoffrey Chase. (He also sent us a nifty Heapsort benchmark which we didn't have time to use yet). SIDES3 will be used primarily to test the speed of a language's function library, on the assumption that if the trig functions are fast, then the rest of the function library also is, since programmers are usually consistently tricky or conventional. Geoff reports the following results:

EDU-20 BASIC, one-user configuration	197	seconds
OS/8 FORTRAN IV	8.8	seconds
FORTRAN-2 with E.A.E. hardware	.27	seconds
U/W FOCAL Version R (carrying 10-place accuracy)	32	seconds

```
SIDES3  3/30/77  EDUCOMP BASIC V2.0

 100 REM.        Prosram to solve S.S.S. (siven 3 sides of a triansle)
 110 !           PAS 3/77          for benchmarkins
 112 INPUT A$
 115 FOR I=1 TO 100                ! benchmark -- otherwise very stupid !
 120 RESTORE
 125 READ A,B,C
 130 IF C>=A THEN 170
 140 T=C
 150 C=A
 160 A=T                           ! SWAP 'EM
 170 IF C>=B THEN 210
 180 T=C
 190 C=B
 200 B=T
 210 REM.        USE LAW OF COSINES
 220 U=C*C - A*A - B*B
 230 U=U/(-2*A*B)                  ! U = COS(ANGLE 'C') opp. to side 'c'
 240 IF U=0 THEN 300               ! Beware of 90 DEGR. (PI/2)
 250 U=SQR(1-U*U)/U                ! U = TAN(ANGLE 'C')
 260 U=ATN(U)                      ! U = ANGLE opposite to side 'c'
 270 IF U>=0 THEN 310
 280 U=U+PI                        ! correct the quadrant
 290 GOTO 310
 300 U=PI/2                        ! this 'BASIC' keeps a permanent 'PI'
 310 A1=U
 320 R=SIN(U)/C                    ! sin/opp. side: constant for trianSle
 330 REM.        Now look at side 'a' and opp. ansle 'A'
 340 U=A*R                         ! U=SIN(A)
 350 GOSUB 530
 360 A2=U
 370 U=B*R
 380 GOSUB 530
 390 A3=U
 395 NEXT I                        ! end of benchmark
 400 PRINT
 410 PRINT "SIDE A =";A," ","SIDE B =";B," ","SIDE C =";C
 420 PRINT
 430 PRINT "ANGLE A",A2;"RADIANS",
 440 U=A2
 450 GOSUB 600
 460 PRINT "ANGLE B",A3;"RADIANS",
 470 U=A3
 480 GOSUB 600
 490 PRINT "ANGLE C",A1;"RADIANS",
 500 U=A1
 510 GOSUB 600
 520 GOTO 700
 530 IF U=1 THEN 570               ! Beware of 90 DEGR. (PI/2)
 540 U=U/SQR(1-U*U)                ! U becomes tansent
 550 U=ATN(U)
 560 GOTO 580
 570 U=PI/2
 580 RETURN
 590 !  ------
 600 U=U*180/PI                    ! to desrees
 610 D=INT(U)
 620 U=U-D
 630 U=U*60
 640 M=INT(U)                      ! minutes
 650 U=U-M
 660 U=U*60
 670 S=INT(U+.5)                   ! seconds
 680 PRINT D;"desr.";M;"min.";S;"sec."
 690 RETURN
 695 DATA 120, 207.846, 120
 700 END
```

```
SIDES3  3/30/77  EDUCOMP BASIC V2.0

?

SIDE A = 120              SIDE B = 120              SIDE C = 207.846

ANGLE A       .5236 RADIANS 30 desr. 0 min. 0 sec.
ANGLE B       .5236 RADIANS 30 desr. 0 min. 0 sec.
ANGLE C       2.09439 RADIANS             119 desr. 59 min. 60 sec.

READY

AROUND 6.6 SECONDS.......(UNTIL PRINTING BEGAN)        ^U
```

Program Notes

Why swap A,B,C?

(a) This is inherited from a test program that did a run-time INPUT. (Kill 112 through 125, write "120 INPUT A,B,C." Also kill 395 and 695, natch.)

(b) The arc cos (effective) is sensitive to quadrant. If C is the largest side, the opposing angle is the only possible obtuse angle and gets trapped rather neatly. Otherwise one would have to move cautiously with the (effective) arc sin — the value returned might need replacement by 180° value, or not. This gets rather nasty.

(c) I have versions of the A.K. Head recursive arc sin and arc cos for BASIC. Slow, but they can be useful. No need to worry about (2h-1)-90°! ■

Shoestring Timeshare

R.D. HAAS

Many schools are caught in a budget crunch and there is fierce competition for scarce funds. Hard decisions usually place computer facilities near the bottom of the priority list, and so there may be little incentive to begin computer oriented curricula. Multiple user systems may be prohibitively expensive for individual schools or small districts. Even if they can tie in to regional or state consortiums or into university-based computer centers, the cost is high. Rural or isolated schools may find that communication lines or toll fees, in the long run, exceed the cost of terminals and user fees. Students and teachers at Clairemont High School in San Diego, California have developed a relatively inexpensive, flexible and expandable multi-user system based on an INTEL 8080 microprocessor.

Clairemont High School has for several years had a single terminal (Texas Instrument 700) hard wired to a centrally located Hewlett Packard 2000 ACCESS system. On a lease-to-purchase plan, the intial cost was about $6,000 for the terminal and a single H.P. port, plus approximately $500 a year in service and maintenance fees.

When BASIC was introduced into the "computer math" class, the need for online computer time made multiple terminals an absolute and imperative necessity. The large initial costs and fees for expansion to 8 or 10 ports was economically impossible at our school. Several alternatives were considered, including multi-plexing several terminals on a single port. A "poor man's multiplex" seemed to be the best interim solution to our problem in the fall of 1975, and so we ordered and assembled a TV typewriter kit (Southwest Technological Products, Albuquerque, NM). This inexpensive terminal (approximately $300) was connected through a simple DPDT switch, along with the T.I. terminal by modem and phone line to the HP 2000.

The TV typewriter has built-in 1K of RAM that allows the operator to compose and edit programs on the monitor screen in a format of 32 character/line * 16 lines/page * 2 pages. The TV typewriter also has a "screen read" facility that allows the student operator to dump the 1K program into the HP in about two minutes while the T.I. stands by, off-line. We ordered a second TVT to be used in the manner described but quickly realized that this system too was inadequate for a large class. The increased down-time of the T.I., the difficulty in scheduling the dump periods, and the inconvenience of the PURGE and SAVE command for each program after each terminal switch operation, along with the accompanying danger of accidentally destroying or mutilating another student's program, prompted us to find another solution.

With the bliss of ignorance we decided to create our own on-site multiple-user system with extended BASIC as our language. In May 1976 we received our ALTAIR 8800 kit with 24K RAM, five serial and two parallel I/O ports and a cassette interface. A group of students in a new pilot course, "Computer Technology", spent the summer session assembling and trouble-shooting hardware while another group worked on writing and debugging software. By the middle of July we were writing simple programs in the 8080 machine code.

We were fortunate to have with us a guiding genius and tireless worker, Jeff Levinski, a student of computer science at Berkeley and a recent graduate of our school. His first important contribution to our system is his assembler-editor-8080 simulator for the HP that makes it possible to assemble extensive programs in INTEL 8080 machine language using the INTEL mnemonic instruction set. His "loader" program is equally important as it dumps the assembled programs directly from the HP into ALTAIR memory where it can be immediately run. We then proceeded to write the necessary bootstraps and other functional operating programs for our system.

In August 1976, Levinski accomplished what we believe is a major breakthrough in micro-software. He modified a commercially available 5K BASIC into a time-share version that we now run regularly on 4 terminals (it can accommodate up to 16). Since September, a few unusually able students have applied Levinski's techniques to modify a much more powerful extended BASIC to use in our expanded system. It is our goal to have eight ports in operation by September 1977. With the completion of a floppy disc drive, we expect to have file capability early in the next school year.

Our total cost, to date, is under $6000, including one ALTAIR 8800, one MITS Floppy Disc drive, 32K RAM and one video modulator board, one paper tape reader, one ASCII key board/encoder, one SWTP printer (40 characters/line), two cassette interface (a KC and a Tarbell) and two cassette recorder-player machines.

We are convinced that with the advent of inexpensive micro-processor systems, of which there are several available off-the-shelf or in kit form, no school should be without on-site multi-user computer facilities. ■

**The author was the sponsor of the recent AEDS Programming Contest winner. He is with Clairement High School, San Diego, CA 92117.*

Reprinted with permission from *AEDS Monitor*, Jul/Aug/Sep 1977.

Talk to our Computer... and it will talk back!

(Plainly speaking, it's only from the Digital Group.)

Now, your Digital Group computer becomes more than a silent partner. You can vocally command your computer . . . it will listen . . . and it will talk back to you. How? With the introduction of the exciting new Digital Group/Votrax Voice Synthesizer.

All this is possible because the Digital Group/Votrax Voice Synthesizer has an unlimited vocabulary, with 64 "human sounds" that can be combined and re-combined to form words and languages. Imagine your own computer glibly spouting English, Latin, Spanish, Russian, Japanese and Yiddish. And 100 average English words require only 1200 bytes of memory!

Programming the Digital Group/Votrax

The Digital Group/Votrax Voice Synthesizer kit is supplied with demonstration and diagnostic software which will permit preliminary testing. Assembler listings of the code involved are included.

We have additional software available at nominal cost:

- "Talking Basic" — $10. MAXI-Basic output converted to English.
- "Talking CW" — $10. For impressing your HAM buddies. Requires the forthcoming HAM interface card.
- "Latin and Spanish Talking" — $10. Hear the computer repeat letters and words typed in Latin or Spanish.
- Demonstration Tape — $5. A sample of audio tape and a complete explanation of the system.

Bonus: A basic input circuit is included that may be programmed to understand a small vocabulary of voice commands.

Unlimited Applications

Consider these possibilities:

- An aid for the blind, with the Voice Synthesizer supplementing a CRT display
- Astronomy — voice input and output of celestial coordinates where light would spoil "night vision"
- Robotics
- Games
- Student terminals
- HAM radio repeater telemetry systems
- Student language pronunciation learning

Let's Talk Price

Actually, we should be shouting this one. The Digital Group/Votrax Voice Synthesizer, with all its capabilities, is only $495 kit or $595 assembled and tested. That's language anybody can understand.

O.K., you've listened briefly to what we have to say about the new Digital Group/Votrax Voice Synthesizer. But we can keep right on talking! Write or call today for *all* the details — music to your ears.

the digital group

po box 6528 denver, colorado 80206 (303) 777-7133

HOW LATE CAN YOU SLEEP IN THE MORNING?

David H. Ahl

Probabilities and expected values are a vital part to writing almost any game or simulation. Here are two real-life problem situations (I face them practically daily) which can be solved with simple statistics. Be warned: the second is considerably more difficult than the first.

GETTING TO WORK

Driving to work, you can take one of two routes. Route 1 is 5 miles long and has 4 traffic lights. Each light is on a 1-minute cycle but with different intervals. Light 1 is green in your direction for 30 sec., red for 30 sec. Light 2 is 20 sec. green, 40 sec. red. Light 3 is 25 sec. green, 35 sec. red. Light 4 is 40 sec. green, 20 sec. red.

Route 2 is 5.2 miles long with only 1 traffic light which is green your way 20 sec., red 40 sec.

Speed limit in the town is 35 mph.

1. Which is the best route and what is the expected time difference between the two?

2. Route 2 also takes you by a factory loading dock. If a truck is just arriving (occurs 1 day out of 30) you will be held up an average of 3 minutes. Does this change your answer?

AND ONCE THERE

Parking your car, you rush into the 10-story Morristown AT&T Building for your 8:30 am appointment with the vice president whose office is on the 9th floor. As you reach the elevators you glance at your watch and see that it is 8:29. From past experience you know that there is a 40% chance of an elevator stopping at any given floor; a stop takes an average of 10 seconds. The elevator passes from one floor to another in 6 seconds. There are 3 elevators all of which have an indicator on the 1st floor (where you are) that shows the location and direction of that elevator. At the point of your arrival, each elevator is equally likely to be on any of the 10 floors going on either direction.

1. Assuming you take the elevator, what is the probability that you will make your appointment on time? What is the probability that you will be less than 1 minute late? Less than 2 minutes late?

2. Under the conditions stated, exactly when would you expect to arrive on the 9th floor?

3. You also know from past experience that you can run up the stairs to the 2nd floor in 10 seconds. The 2nd to 3rd takes you 10% more time (11 sec.) and 3rd to 4th 10% more time (12.1 sec.) and so on. What position of elevators upon your arrival would cause you to run up all 9 floors? (Many answers are possible — select one "break even" combination).

4. Assuming your office is on the 3rd floor and you are faced with the same situation as above (time 8:29 with 8:30 appointment on the 9th floor), but with no elevator indicators, what is your best strategy to be on time for the meeting or as little late as possible? That is, do you run up or wait for the elevator?

PARTIAL ANSWERS

"Getting to Work." 1. Route 2 is approximately 0.47 sec. faster. 2. The arriving truck has an expected value of -6 sec. per day on Route 2, hence Route 1 now has the edge by 5.53 sec.

"And Once There." Once you get on an elevator, it is a fairly simple matter to determine how long it will take to get to the ninth floor (8 floors x 6 sec. = 48) plus (40% chance of stopping x 10 sec. per stop x 8 floors = 32) equals 80 sec. But figuring when the elevator will arrive on the first floor is something else again. Any elevator can be at any floor going in either direction. Hence, Elevator 1 has a 0.056 probability of being at, say, Floor 3 going up*. How long until it returns to Floor 1? Well, if we know it goes to the 10th floor, that's easy, but it has only a 0.4 chance of going to the 10th floor, 0.4 chance of going to 9 and so on. Multiplied by 18 different possible starting positions and by 3 elevators, this is a nasty problem. In a situation like this you have to ask yourself whether a heuristic, or rule of thumb, or best guess wouldn't provide an adaquate answer. For example, you might want to make the assumption that at least one elevator is at the 4th floor (or below) and heading down.

It is sometimes easier to come up with a solution if you think of the problem in entirely different terms. For example, think of the elevator as a one-way trolley on a circular track — station 1 is Floor 1, station 2 is Floor 2 going up, and so on. Station 18 is Floor 2 coming down and then back to station 1 again. Using this approach *may* make it easier to work the problem.

By the way, it should be apparent that a computer isn't much help in solving this particular problem. However, if this were part of a much larger simulation in which the output of one part provided the input to the next (a very typical situation), a computer would be almost vital to the solution.

If you're still with me and want to read a fun little book on the subject, get "Flaws and Fallacies in Statistical Thinking" by Stephen Campbell published by Prentice-Hall.

*If there are 10 floors and the elevator has an equal chance of being at any floor going in either direction, why isn't the probability of being at Floor 3 going up 0.1 x 0.5 = 0.05? Simply because Floor 1 and 10 do not have a direction associated with them, hence there are really only 18 locations. Actually, that's over-simplified because the elevator may not even reach Floor 10, or 9 or 8 etc. ■

Featuring on-the-spot interviews with two manufacturers, two retailers, and a publisher

Gamboling in Atlantic City

David H. Ahl

Atlantic City: Gambling has been approved, although it will be another nine to twelve months before it gets under way. Nevertheless, the beaches were a little bit cleaner than last year, the boardwalk traffic a bit brisker, and lots of construction was underway. On the other hand, the Shelburne Hotel, site of the "Personnel Computing '77" was as seedy as ever, the air conditioning in the exhibit hall spasmodic, the elevators generally out-of-order, the help surly and unpleasant. So the convention flopped? Wrong! Through the extraordinary efforts of John Dilks, Dave Jones, their wives, and other organizers, PC '77 was a fantastic success!

Creative sponsored a wine and cheese reception for the 400 or so conference and exhibitor people on Friday night. We poured the last people out around 3:00 A.M. Nevertheless, most people appeared for duty in their respective booths at 9:00 A.M. on Saturday. And a good thing too. All day Saturday and Sunday, virtually every exhibit was five-deep with people. The workshops and presentations were all crowded to overflowing. Anyone who went and said they did not have the experience of their life was either lying or asleep.

Heathkit showed their lines of computers in public for the first time. Commodore had a bunch of Pets. MITS/Pertec had a huge, showy booth. SWTPC had disk systems. OSI showed their hard disc. TDL had Xitans, Cromemco had Z-2s with built-in disks, Midwest Scientific had their new computer, and on and on ... Even Teletype was there.

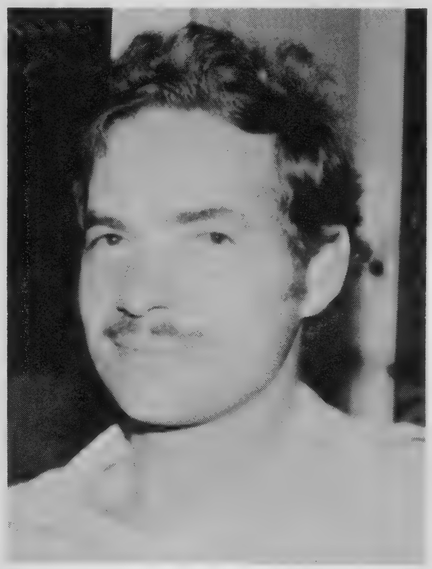
In the bedlam, I managed to talk to a cross-section of industry notables: a publisher, two manufacturers, and two retailers. Lightly edited transcripts of these conversations will be found on the next few pages.

Next year the exhibits will be in Convention Hall and some of the new hotels will be finished. So, put PC '78 on your calendar now ... the last weekend in August. It will be dynamite!

Adam Osborne

Adam Osborne is founder and president of Osborne and Associates, Inc., in Berkeley, California. He is a consultant on microcomputer systems, and author of the popular books, *An Introduction to Microprocessors, Vols. 1 & 2.*

Ahl: In talking to people here in Atlantic City, I've been trying to get some kind of consensus of where the hobby is going from here. Lud Braun felt that the days of the hobbyist are over and the days of personal computing are coming in with the advent of the Pet and the Radio Shack computers. In other words, home computing is kind of paralleling the early days of amateur or ham radio. From 1900 to 1920, there were only amateurs; then, all of a sudden, commercial radio appeared and brought with it explosive growth. The amateur hobbyists stayed around but had relatively modest growth compared to commercial radio. Do you think this will happen with personal computing? Where do you see things going?



Adam Osborne

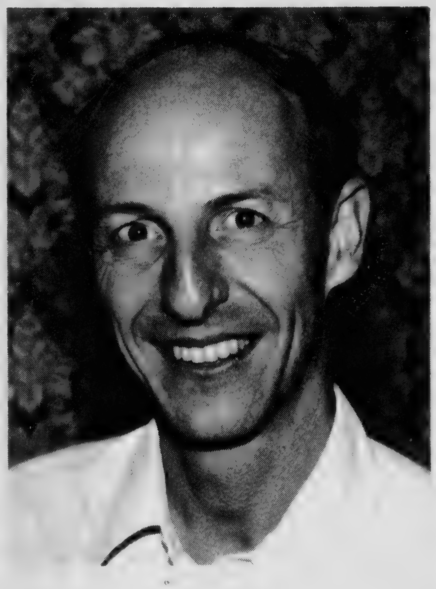
Osborne: Personal computing, in which people have a computer for the sake of being able to program it, is something which will always be difficult to do, so it is never going to become a mainstream thing. It is never going to reach the levels which would interest Sears or J.C. Penney.

Ahl: Do you feel that applies with personal computers such as the Pet or Radio Shack machines?

Osborne: Anything that a guy has to program himself is always going to have a relatively limited audience. Now, when I say relatively limited, it can still amount to two or three hundred thousand people across the country, but this is generally insignificant. There are going to be computers going into the home in a form that you do not have to code. So, if whatever the computer is going to do for you is already programmed, that could run into the millions. You say, well, that is not the same thing, but I argue that eventually that is the way all computers are going. Most mainframe machines will reach the point where you won't have to program them; You will just have to define what it is you want them to do, thus ultimately reaching a merger of the home-appliance-type computer, which you don't have to

program, with the mainframes which you also don't have to program.
Ahl: Does that imply different kinds of languages, or will everything be almost a black box from the user standpoint?
Osborne: Eventually there will be no languages. You will be programming by example. You will create on the screen an image of what it is you want it to do, and the extent to which there is any programming language will be the manner in which you define the variables. In other words, there can be no ambiguities; you must be given some means of identifying and resolving ambiguities. That is the extent to which there will be programming languages. But that will probably be 15 to 20 years away.
Ahl: That sounds like a long while. What is coming sooner?
Osborne: In the meantime you are going to see the more traditional programming languages around. I do not think you will see new ones very much. You will see them come, but I don't think you will see them catch on, because what will eventually wipe them all out will be this concept of an easily-programmable computer.
Ahl: Well, there are things like the high-level language that goes with, say, the Software Technology Music System. It is very easy to learn. It seems to me that there would be a lot of things like that, small languages for special-purpose applications.
Osborne: Yes, but that is almost beginning to be the programming example that I described. You write music in a form on the screen, and that creates for you the programs which generate the music. That is the beginning of the approach. It is not really a programming language any more than writing English is a language. Writing music notes is a language and what they are doing is writing music notes and then letting the machine recognize and play them.
Ahl: What kind of significant hardware do you see coming out in the next five years or so?
Osborne: I think you are going to see very inexpensive and very complex electronics. You are going to see the LCD display, the liquid crystal display replacing the touch switches, replacing the keyboards. At the end of the five-year period we are going to see great pressure put on floppy disks and rigid disks by memory devices ... such as CCD, bubble memories and other non-volatile types. Perhaps a little later, we are going to start seeing some real impact on printer technology from laser technology. It's expensive today, but in five years it's going to start becoming quite inexpensive. The laser creates a static charge so that a dry ink can be deposited on it.
Ahl: Like xerography?
Osborne: A xerographic-type approach. Xerox is doing that right now but the price is high and the quality relatively low which is the way all these things start. Then the quality becomes high and the price becomes low.
Ahl: Where do you see your role, Osborne and Associates, in the future?
Osborne: I think there is going to be an ever growing need for a link between the technological side of society and John Q. Public who has got to live with that technology, and that is the need we are going to continue to address. We are going to be the translators; we are going to allow the public to understand what they must understand in order to live successfully in a technological society. For example, right now, computers have run wild in our society for the simple reason that no legislators understand computers well enough to draw up sensible laws, and the public at large will accept anything they are told because they don't know any better. I want to curb that. In the future we are going to carry on in that direction, to inform the public in terms that they can understand as to what is going on in the technologies which are affecting their daily lives.
Ahl: Do you see yourself staying mostly in the publishing business or branching into other means of communicating?
Osborne: I see myself basically staying in the publishing business — because right now that is the effective low-cost medium. We have not yet reached the point where video cartridges or anything else is nearly as good as visual graphics. It is going to be some time before any new technology comes along to replace the book.
Ahl: Adam, thank you very much for your thoughts and ideas.

Robert Suding



Robert Suding

Robert Suding is founder and president of the Digital Group in Denver, Colo. Digital Group Products have had an image of being "different"—non-standard bus, large circuit cards, naked hardware (in the beginning), etc. But one other thing stood out—high reliability, hence assuring the Digital Group a place in this rapidly expanding market.

Ahl: In your own perspective, where do you see the home-computer field going in the next couple of years?
Suding: Well, I see them probably trying to define more definite applications. I see a trend away from games, other than just for beginners

The exhibits were mobbed from Saturday at 9 AM to Sunday at 5 PM.

RCA showed their new COSMAC VIP microcomputer.

and looking to more serious applications. One field that I am trying to get a number of people directed into is using microprocessors for handicapped people. So I am starting a new organization which will be amateur-computer fans designing and building and implementing projects for handicapped people. We have also set up sessions for various conferences next year showing how the amateur computer fans have implemented things for handicapped people.

Ahl: What specific things are you working on?

Suding: We are working on voice synthesizers for blind people that will allow a blind programmer to have a self-sufficient job using a voice terminal instead of a video terminal. We are working on devices to allow people with cerebral palsy to communicate through specially programmed input or output devices which could speak to them or it could type for them. It is something that is not of any one manufacturer, but something that almost any one of the various hobbyist microprocessors are far more than capable of doing.

Ahl: I heard you mention before that the Digital Group is planning to offer a speech synthesizer. How will it compare to some of the others that are available today?

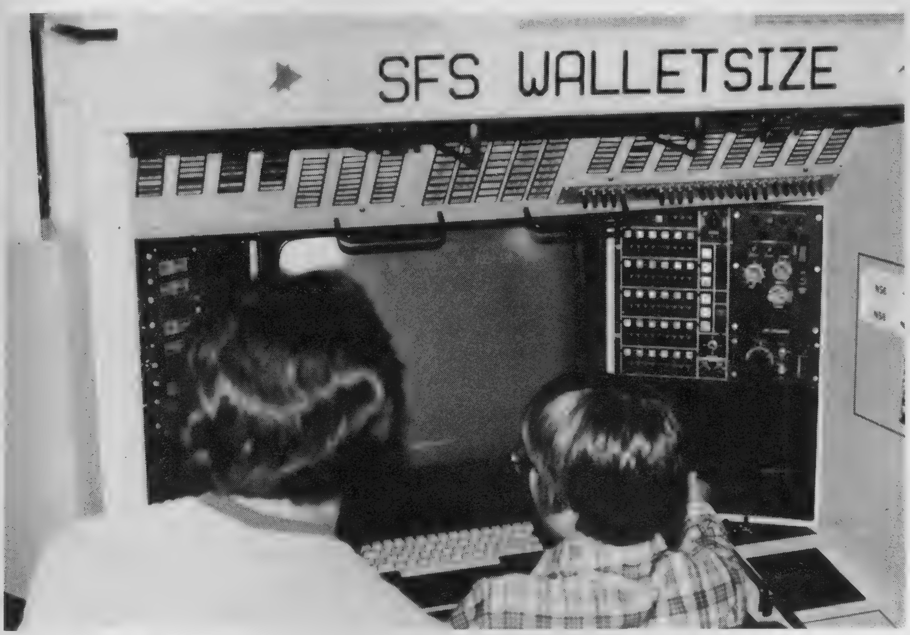
Suding: The main thing about our speech synthesizer is that it uses very low programming overhead; it takes roughly 1200 cycles for every 100 words spoken. The reason for this is because it has a relatively limited total flexibility; it has no inflection and it cannot sing. It does not do anything but produce 64 different human sounds. The job of the microprocessor is merely to link the 64 different sounds into the desired language whether it be Russian or Spanish or English or Latin or any other language.

Ahl: Is it likely to come out with that strong Swedish or Slavic accent so characteristic of the earlier Votrax units?

Suding: Yes, this will definitely have more of a Swedish accent than most here. I heard some interesting rumors on how that originated; I suspect they are not true any more than any other rumor in hobbyist circles. I understand that it was invented by a Swede in Wisconsin, who developed the original algorithms for speech synthesis in the way he thought English was spoken.

Ahl: What other kinds of things, other than in the area of the handicapped, are you working on, or do you see being in the future?

Suding: Well, I am personally interested in energy management in homes and I am personally designing a system for my solar-energy house in Colorado where we actually control the whole house under process control. Existing solar systems are typically implemented using just standard relays and hard logic and temperature-differential thermostats. The microprocessor introduces a rather new element, and that's the possibility of programming the thing in the language of the homeowner, making it possible to use a "home management" language as opposed to Fortran or assembler and so on. The person talks to the thing and interacts with it by the machine asking how many watts he has in the system that he wants to control, what temperature he would like in the various rooms. He will be able to custom-program the house if he has visitors, or if there is a party; he can set it up to take into account various differences at his convenience. It offers some tremendous possibilities for energy conservation because it greatly enhances intelligence and record-keeping. I will be able to realistically evaluate different overall systems for using the minimum amount of energy in my house. I project that the cost of this processor will be quite a bit higher in my case, since I will have a lot more power, but a processor for a "normal" home controller being built now will be



2005AD of Philadelphia surrounded a three-screen video display running LUNAR LANDER with all sorts of real cockpit controls and had continuous lines of kids waiting to get at those controls.

about $300 or $400 quite easily.
Ahl: What about all the sensors required? Won't they be a lot more costly than the processor?
Suding: Probably not, because there are several new sensors recently announced in the one-dollar range. You also have some older temperature thermisters or transducers in the range of about $3 to $4 surplus that are very excellent. By using thermisters and analog multiplexors and something as simple as additional voltmeter chips and some old surplus telephone cable, we could probably wire up a complete house with the temperature input system and multiplexors in the range of about $100 for all the equipment; it all depends on how sophisticated you want to get. And then for control you can have spot room control and use little butterfly dampers that are run by a geared-down motor which you can buy surplus for about $3 or so and modify standard ductwork to have all this programmable. You should be able to justify the cost of the whole system in a year or two in what you save in heating bills.
Ahl: It sounds like you have to be a pretty good mechanical, electrical and civil engineer to implement such a system today by yourself.
Suding: It certainly helps but you would be surprised at what can be done. The biggest problem is finding people who know something about heating and solar and thermodynamics and who also know programming and computers and systems design; you can imagine these people are somewhat hard to find. That is why I say probably the best solution would be to develop a training language in terms understandable to a heating construction contractor.
Ahl: Do you find today that most of the customers of the Digital Group are relatively more knowledgeable hobbyists than in the past, or are they businessmen or what?
Suding: Much more knowledgeable hobbyists. The typical hobbyist no longer is one that will buy anthing. He is very, very knowledgeable. He has friends who have recommended things; I believe most systems are sold through a person who has a friend who has a system like it. We continually found, especially in the earlier days, we would send a system out to a given geographic area and within a couple of weeks we would get a couple more orders from that same area. It was obvious that the person building the system, especially if he got it running immediately, would call over a friend to watch it and he would be so impressed he would send in an order. It's a major investment and people just don't like to make major investments without a fair knowledge of the company and the product. We find that our biggest problem now is not being able to meet the demand; I suppose I can rationalize that it's a stabilizing influence in that it smooths out production.
Ahl: Well, there certainly are worse problems!

Daniel Meyers

Daniel Meyers

Daniel Meyers is founder and president of Southwest Technical Products Corporation in San Antonio, Texas, SWTPC originally made high-quality low-cost hi-fi amplifiers and other electronic kits. In the fall of 1975 they introduced the SWTPC 6800 computer kit in the same tradition and it has seen a wild, explosive growth ever since.

Ahl: I see you have some disk-based systems here and also that your components have nice pretty covers on them. That is a change for SWTPC!
Meyers: Well, one of the features that people had been asking for on the old box was a cover. So we were nice to them this time. We went ahead and put the cover on. We are doing what everyone else is doing, of course. We are expanding our systems, we have disks now, we have terminals, we have a printer and lots of new little inner goodies to go in the box, we have an interrupt timer now, we have a calculator interface, we will have an A-to-D, and also a PROM programming card that goes in the interface slot.
Ahl: The A-to-D, how many channels will that have?
Meyers: I don't remember, 6 or 8. It's designed around the National chip.
Ahl: Will you put some hooks in Basic to support these peripherals?
Meyers: Oh, yes.
Ahl: I notice a lot of companies have come out with grand schemes and pieces of hardware and then no way of using them, even with their own software.
Meyers: No, no, we definitely know about that need and we are going to take care of it. We will try to make the things useful to people. That was the philosophy behind, for instance, the little interrupt timer we recently introduced. A lot of people have oscillators that you can hang in their box, but you have to slip a switch to get different timings. OK, with ours you

Commodore showed five real, live PETs.

John Mauchley leaves banquet attendees (including John Dilks) with some words of history and wisdom.

don't do that. Ours has a software programmer at the divider module and you can get anything from a microsecond on up to an hour, software-controllable.

Ahl: Can you use it in high-level software, such as BASIC?

Meyers: Yes, you have to write it in using the instructions that go with that timer, in a user-defined machine language subroutine, but that's provided for in our BASIC. All of that is described in BASIC and the interface, of course, has complete instructions on what you want to tell that thing. If you wanted to change your timing intervals in the middle of the program, you could. It's very useful!

What else are we doing? Well, of course, we are trying to stay one step ahead of the Radio Shack wolf.

Ahl: Do you see Radio Shack and Commodore as being threats, or enlarging the industry and making more of a market?

Meyers: Both. Radio Shack could be a definite threat but Commodore, I don't really think so.

Ahl: Why is that? Why do you differentiate the two?

Meyers: Well, if you look at them, the Commodore apparently is very difficult to add on to and to interface with external devices; it costs more to expand the memory than our things do, or anyone else's for that matter. $300 or something ridiculous like that, is what I heard to add 4K. And the keyboard is unusable for touchtyping. Anyone who bought it should be happy with it for at least a week, before they wanted something better. Radio Shack, unfortunately for companies like us, has a pretty good product. It's got a creditable keyboard, it can be expanded quite easily, and it could definitely be a very competitive system.

Ahl: Although by using a non-standard bus, it may limit the number of peripherals.

Meyers: However, if they have the peripheral available, the purchaser will buy it. And that's just the point: they apparently intend to get the peripherals, so we'll see on that one, I don't know. Commodore, I feel, will get some people interested in computers, but they won't be much of a competitive threat as far as what we're selling.

Ahl: How do you see Heathkit?

Meyers: Heathkit has said they're not going to sell theirs assembled; they sell them by mail and through their stores, of which there are a limited number. We intend, by the end of the year, to be offering assembled units. I don't think that anybody can sell kits for business applications; it's just not meeting the need.

Ahl: You said business applications. Do you see more of your machines going into that kind of use?

Meyers: Well, I hope so. We're working towards that. We feel right now that we have the disks; we manufacture really everything that you need for a reasonable business disc system other than a printer and we're looking for one of those, but we haven't found the right mechanism yet. But that's the only item we can't supply currently. We are also working with a guy in California who has developed an excellent, very fast BASIC compiler. BASIC is lovely, but BASIC interpreters are ridiculously slow for business. Anyway, we have the disks and we have the language at an adequate speed, we have a more-than-adequate computer, so I think we're close.

Ahl: What about support in the business market? Don't they need more hand-holding and support? Do you expect your dealer network to be able to provide that?

Meyers: Some of them can, some can't. The fact that the language is BASIC is a big help in my opinion. There are a lot of application programs available now in BASIC. The first people, I think, that are really going to use these things in

Dazzle doodle (electronic Etch-a-Sketch) was a continuous attraction at Creative's booth.

business are the ones that have been here looking; the lawyers, the doctors, the accountants, the ones that understand electronics and are interested. Later you get to the corner ice-house and those places where you have to walk in and set it on the counter and say, "This is a diskette and it goes here, and you push B and the system starts," and so forth. It's going to be slow. I don't agree with any of these people who say we're right on the verge of this huge market, and it's going to grow into this colossal industry in a year or two....

Ahl: Some people have said that this is the year that will separate the hobbyist market from the home and small-business market, the hobbyist is essentially finished, and future expansion will come from business systems.

Meyers: There's only one thing wrong with that. My opinion is that *nobody* at this show has got a system with the possible exception of MITS/Pertec that could be used in a small business. Period. I just don't believe I've seen anything else here that would be suitable, because they're lacking either the language, or reliability, or something. Any one of them you want to name has got a problem somewhere that is going to keep them from being successful in selling their system to businesses. And they don't really understand that. I think *I* understand that because *I've* used a computer in *my* business for 3½ years now and I know what my problems are, I know what I've got to have, and I don't have a system that I could use in my plant yet.

Ahl: I have to tell you this. We have five computers at *Creative Computing* and I don't use a single one of them for the business.

Meyers: Right. We're just not quite there yet. But we're on the verge of being able to sell a system that would work very nicely in a smaller business. I don't think anybody here today, with that one possible exception, could say that they have an adequate system for small business, so we haven't even started getting into that. It's going to be a long, rough road. If that's where people intend to make their money, I think they're going to be in for a big shock when they try.

Some of the Creative Computing barbeque set, l to r: Alan Salisbury, Burchie Green, Linda Eckerstrom, Steve Gray, Sandy Ahl.

Alan Hald

Alan Hald is founder and president of Byte Shops of Arizona and Byte Shops Mail Order. The Byte Shops franchise has had a rocky history, but Alan's store and mail-order operation with its outsize, popular catalog has clearly been a success right from the start.

Ahl: The first question is: can the Byte Shop of Arizona find happiness in New Jersey? Seriously, why did you decide to exhibit here?

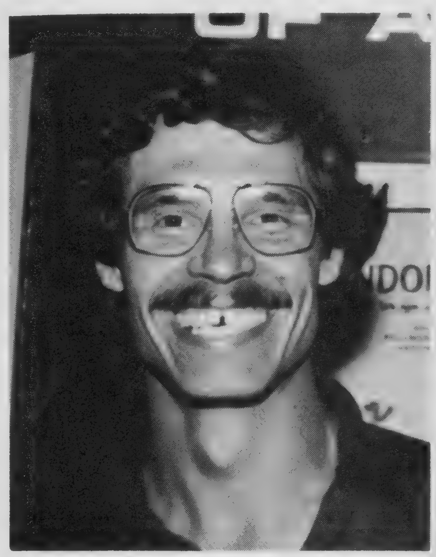
Hald: We have Byte Shops Mail Order which is nationwide, and we have a lot of customers on the East Coast. The Byte Shops of Arizona run the Byte Shop Mail Order. We also handle Micro Age which distributes to a lot of independent small stores and a lot of those customers are also here on the East Coast.

Ahl: Where do you see things going from here in the next year or two?

Hald: It's hard to see our two years, but what we will see next year is the beginning of full-fledged systems stores. We are going to be involved in systems for various businesses and occupations, and a continuation of the growth of the hobby stores, and the beginning of the mass retailing of low-cost systems. However, I think it's difficult for a lot of the stores to get involved in the mass market mainly because of the demand and the way the companies like Commodore deal with the products. They are looking for the very-large-volume buyers.

Ahl: Do you see Commodore and those products being a threat to the independent stores?

Hald: No, I think what will probably happen is they'll open up the market even wider than it is now. It is difficult for us to appeal to a wide range of the public when you are talking about systems that cost $1,000, particularly assembled systems. There is a tendency now for people to want assembled systems rather than kits. Once that price barrier is broken, there will be more people coming into the market. They will take one home and use it, and once they start learning about computers, they will want one with more capability. So people will become more intrigued with adding on to their systems or exchanging systems and upgrading them. They will probably be disappointed to find they cannot add anything on the Pet like a disk drive and they will come in and look at a North Star and see how convenient it is to use, and they will start looking at S-100 types of systems. But I think if someone like Commodore is very successful, maybe other larger corporations will say, "If they can do it, we can," and that, in effect, will open up the distribution channels. If that is very successful—

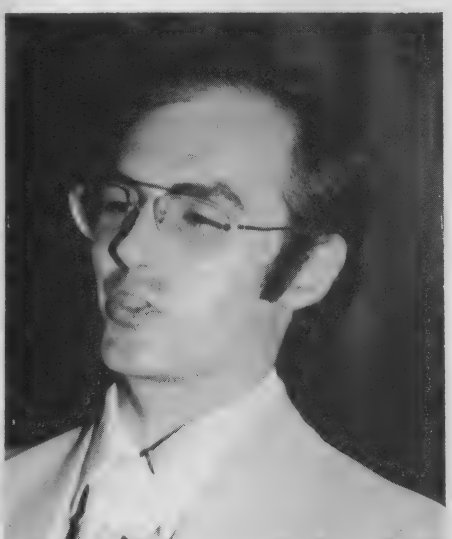


Alan Hald

let's say Commodore hooks up with a large distributor like Sears—then someone like Sharp will come in and wonder who else to distribute to, and I would say the computer stores are a natural since they are the hobbyist place and they will be hungry for products.
Ahl: Well, I have noticed that a number of the computer stores today are carrying those little Microchess sets, as well as the Fairchild games and other prepackaged products. I see a kind of a cross-fertilization there.
Hald: Very true, what is happening now with product lines becoming so broad is that a lot of stores are going to have to start making decisions as to what type of store they are going to be. The capital requirements to support a store are limited. Some stores are moving towards systems and they don't want to be bothered with hobbyists; what they are trying to sell are the big systems. On the other side, some will develop into more consumer-oriented stores that handle shelf games, hobbyist type of computers, and the kits. I think we will see two types of stores emerging, and probably eventually a third, that being a mass-consumer type of store. It will be interesting to watch.
Ahl: It sure will!

Ken Greene

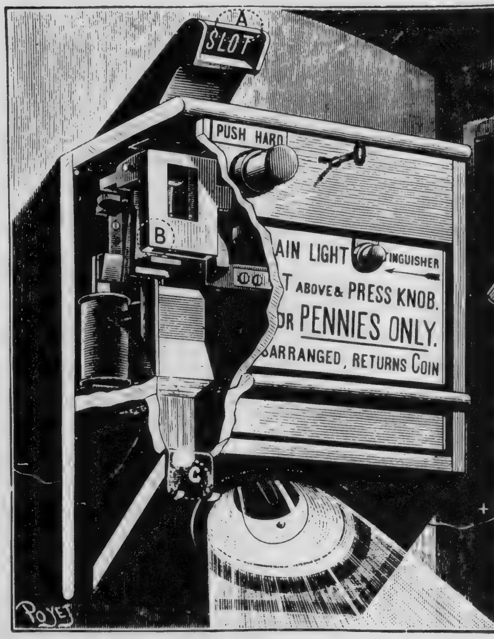
Ken Greene left a steady, well-paying job in industry in early 1977 to open a computer store in Morristown, New Jersey, the first of the ComputerLand franchises. He works twice as many hours a day now, but wouldn't give it up for the world.

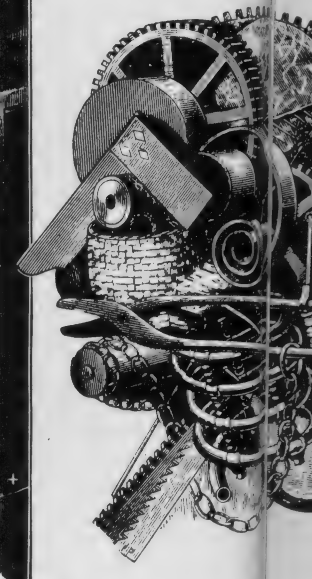


Ken Greene

Ahl: What do you think of the show, Ken? Have you gotten business, or references? How is it of value to a retail store?
Greene: Well, we make contacts with people in our area who did not know we existed, and vice-versa. I believe the show is a really good place.
Ahl: What kind of people have you been talking to here compared to the kind of people who come into the store?
Greene: A lot more business-oriented people here than come into the store. There you get many more hobbyists. Here, people are looking for packages; they want an inventory package system, or some other business system. Big systems too, like $10 to $12 thousand.
Ahl: You mean here! At the show?
Greene: Absolutely. Very, very few people came by who were hobbyists. They may have come too, but they didn't talk. They came though and they said, "Gee, that's nice" and then left.
Ahl: That's interesting, because you would think that a show like this which was billed mostly as a hobbyist show, would not draw the business-type customers.
Greene: Well, I think next year it won't be anywhere near a hobbyist show. I think next year, we will see even the big business systems. I think we will see DEC in here with systems and all the others. Just from what the people are asking for. When we come down next year, we'll have six booths reserved. We may go with even more. Two of those booths will be my separate corporation which will display its business, hardware, software packages; the rest will be my store and other ComputerLand stores.
Ahl: In your own opinion, where do you think things are going in the future?
Greene: Everything looks like its going commercial—$500 Fortrans that require disks. Everybody wants a Fortran but the hobbyist isn't going to buy that. I think the beginning hobbyist is going to get sucked in by the Commodore Pet and the Radio Shack and the Heathkit. And what does that do to a computer store like mine? Well, those companies are just going to educate the market place. The only ones I consider high-line quality units are the Heathkits. And Heath will always command the hobbyist in the build-your-kit market. The computer stores like myself and the Computer Marts and so on are going to be more like a high-level stereo dealer. The guy who really wants to buy the big component system is going to come to us, rather than go to the department store or his local Radio Shack and buy their assembled units. So the future for us is mostly with small businesses and top-line hobbyist systems. ■

Computer Hist





1. The use of algorithms is fundamental to developing computer programs. When was the first known use of the floating-point algorithm, which included conditional branches and iterations?
a. Before 1940. b. 1940-1945. c. 1945-1948. d. After 1948.

2. Before the digital computer age, astronomers had to calculate orbits of astronomical bodies by hand. What was the typical precision of these orbital calculations?
a. 2 places. b. 6 places. c. 9 places. d. 100 places.

3. The government funded development of a computing machine, but cost overruns of more than ten times the original estimates caused the administration to withdraw in alarm. An oft-repeated story, but who was the developer the first time it happened?
a. W. Burroughs. b. N. Dodge. c. C. Babbage. d. G. Boole.

4. The first wholly key-operated calculating machine was the Comptometer, a practical adding and listing machine. Who invented it?
a. Leo Evans. b. Norbert Dodge. c. Francis Galton. d. Dorr Felt.

5. The American Arithmometer Company patented the first practical recording adding machine in 1892. Today the firm is known by another name. What is it?
a. Burroughs Corp. b. Honeywell, Inc. c. Sperry Rand Corp. d. NCR Corp.

6. Herman Hollerith organized the Tabulating Machine Corporation which grew into IBM. His punched cards were first used by which government agency?
a. Federal Bureau of Investigation. b. Census Bureau. c. Internal Revenue Service. d. Department of Defense.

7. Leonardo Torres y Quevedo's "telekino" was an early attempt to program a machine to play a popular game. What was the game?
a. Chess. b. Checkers. c. Tennis. d. Bowling.

8. The application of computers to astronomy was revolutionary. Ephemeris data calculated by Wallace Eckert in 1951 is still in use today and was the basis for many NASA missions. Which of these bodies was in Eckert's data?
a. Jupiter. b. Mercury. c. Earth. d. Bardot.

9. The "Universal Computer" designed by Alan Turing can solve any mathematical problem.
a. True. b. False.

10. Weather predicting may sometimes be no more accurate today than when granny's aching bones signalled a storm. The first person to use data processing in meteorological prognostication got his name attached to a basic weatherman's parameter. Who was he?
a. Zucker Kurtz. b. Lewis Richardson. c. Lenwood Walters. d. Charles Precipitation.

11. Plankalkul was an attempt in 1940 to?
a. Make tires from garbage. b. Develop a new memory device for computers. c. Write computer programs. d. Invent a breakfast pastry.

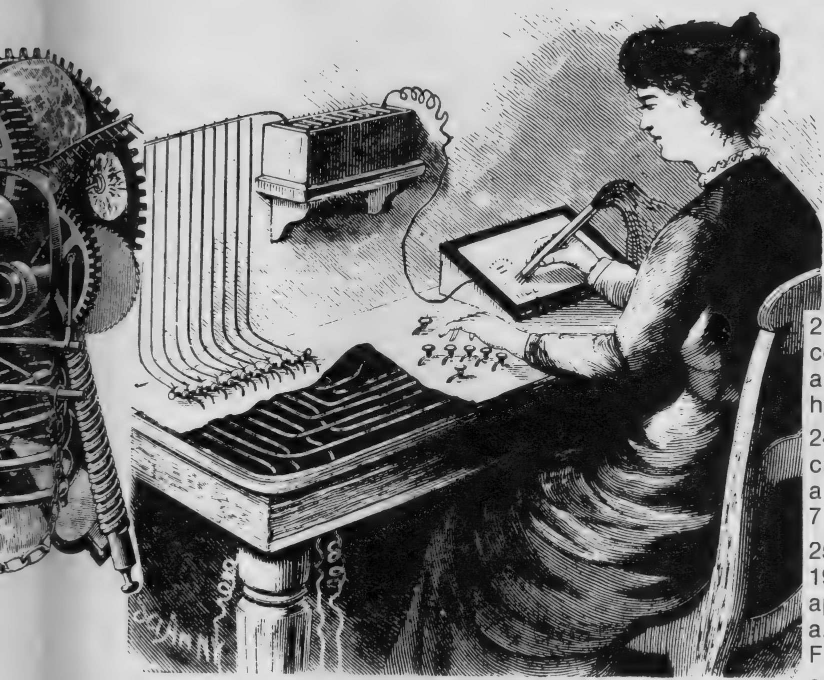
12. Vannevar Bush, the man who coinvented a differential analyzer with Samuel Caldwell also?
a. Discovered a cure for polio. b. Was the military R&D Director during the Manhattan Project. c. Developed the transistor. d. Sold botanical specimens door-to-door.

13. Cybernetics, which is used in control theory, automation, and computer programming was a word coined by?
a. B. Masterson. b. S. Curuthers. c. S. Morse. d. N. Wiener.

14. It was 51 feet long and 8 feet high, tipped the scales at a dainty 35 tons, and used 3,000,000 connections for its 500 miles of wire. Its name?
a. Mark I. b. King Kong. c. ENIAC. d. Jaws.

15. A government-financed machine was developed in conjunction with the Morse School of Electrical Engineering at the University of Pennsylvania. The coinventers were?

tory Trivia Quiz



Edward Pasahow

a. Martin & Lewis. b. Mauchly & Eckert. c. Wells & Fargo. d. Aiken & Wiener.

16. In the genealogy of data processing machinery, the younger brother (sister?) of BINAC was?
a. UNIVAC I: b. IBM 701. c. EDSAC. d. GENIAC.

17. UNIVAC stands for?
a. Universal Alternating Current. b. University of Alta California. c. Unit Versatile Automatic Computer. c. Universal Automatic Computer

18. The conclusion of the judge in the patent infringement suit between Sperry Rand and IBM in 1973 was that the inventor of the computer was?
a. John Atansaoff. b. John Eckert and John Mauchly. c. All of the above. d. None of the above.

19. Which university gave up the fulfillment to many future fund drives when it failed to obtain patents on early computer work done there?
a. Slippery Rock. b. MIT. c. Washington State. d. Iowa State.

20. With its 40 racks of equipment and 20,000 vacuum tubes, ENIAC ran up daily electric bills of?

a. 1¢ (during the war, electricity was free). b. $30. c. $60. d. $100,000.

21. Jay W. Forrester, an electrical engineer and management expert who applied computer simulation to the real world, also?
a. Devised COBOL. b. Retired on the proceeds at age 23. c. Founded the RAND Corporation. d. Invented the random-access memory cores.

22. The stored-program machine, program labels and symbolic addresses, macros, and microprogramming all received early attention of Maurice V. Wilkes at which lab?
a. Oxford University. b. U.S. Army Abeerdeen Proving Ground. c. Dr. Faustus. d. University of Cambridge.

23. The earliest method of storing programs in a computer used tanks containing?
a. Mercury. b. Water. c. Chicken soup. d. Liquid hydrogen.

24. Dr. Grace Hopper was involved with which commercial effort?
a. IBM 701. b. Business programs for the Burroughs 7B. c. UNIVAC I. d. HAL.

25. The first programming language was developed in 1952 for UNIVAC I and was used for numeric and scientific applications. It was?
a. FORTRAN. b. Short Code. c. Macro Language. d. Fuzz.

26. The first commercial, large-scale binary computer used the well known (by ancient, moss-covered programmers) Speed Code. The language made the single-address, fixed-point computer appear to be a three-address, decimal, floating-point computer with index registers. The computer was?
a. IBM 701. b. UNIVAC II. c. Burroughs 102. d. CDC 33.

27. Arguments on optimal storage structures have been going on since the early 50s. Match the machine with its storage technique.
a. UNIVAC I. b. IBM 702. c. Blocked records on tape. d. Unblocked records on tape.

28. The man who thought of "THINK" was?
a. A.A. Michelson. b. W.S. Burroughs. c. T.J. Watson d. S. Craig.

29. Wassily W. Leontief was awarded the Nobel Prize for his computer-associated input-output analysis. His work addressed?
a. Computer communications with peripheral equipment. b. Grocery store POS terminals. c. Economics. d. National defense.

30. In the top 50 data-processing industrial companies, one is a giant. IBM's share of last year's total DP revenue pot was about?
a. 25%. b. 50%. c. 60% d. 75%

The answers are on page 65.

TO BE CONTINUED...

Quiz Answers

1. *a.* The Babylonians during Hammurabi's dynasty (1800-1600 B.C.) developed such algorithms for excavations, linear equations, and geometric problems.

2. *c.* Astronomical calculations were carried out to nine digits of precision before 1900.

3. *c.* Charles Babbage allowed the cost of his difference engine to escalate from £1,500 to £17,000 after eight years of government support.

4. *d.*

5. *a.* The Burroughs Adding Machine Co. came into being in 1905.

6. *b.* Use of punched cards cut the time required for the 1890 census to one-third that to the previous one.

7. *a.* Quevedo's machine was an early effort in the cybernetics field (even before cybernetics had a name).

8. *a.* Ephemeris data on Jupiter, Uranus, Saturn, and Pluto from 1652 to 2060 was generated.

9. *b.* Turing proved that a fixed, definite process on an automatic machine cannot solve every mathematical problem.

10. *b.* The Richardson Number relates gradients of temperature and wind velocity.

11. *c.* Karl Zuse, developer of Plankalkul, also designed and fabricated a family of computers in the 1940s, called Z1 through Z4.

12. *b.* Bush served as R&D Director during much of WWII.

13. *d.* Wiener was interested in all aspects of communications and control in living organisms and machines.

14. *a.* The Mark I, an electro-mechanical computer, was built for the U.S. Navy Board of Ordnance by Howard Aiken and his combined IBM and Harvard team.

15. *b.* ENIAC was produced for the U.S. Army Ordnance Proving Ground for ballistic calculations.

16. *a.* ENIAC begat BINAC and BINAC begat UNIVAC I.

17. *d.*

18. *c.* The judge concluded that Eckert and Mauchly derived some of their ideas from Atanasoff, partly as a result of Mauchly's visit to Iowa State in 1941.

19. *d.* See 18.

20. *c.* 20,000 vacuum tubes use a lot of watts!

21. *d.* Forrester also participated in constructing the general-purpose Whirlwind I computer.

22. *d.* Wilkes directed the mathematics laboratory at Cambridge and was the first president of the British Computer Society.

23. *a.* Mercury tanks formed the ultrasonic memories of the EDSAC machine.

24. *c.* Dr. Hopper was working for the Eckert-Mauchly Computer Corporation at the time.

25. *b.*

26. *a.*

27. *ad, bc.*

28. *c.* Thomas Watson Sr. dreamed up the slogan while working as an NCR salesman.

29. *c.*

30. *b.*

All about recursion and various other goodies offered by APL and even some versions of BASIC.

Something is Missing . . .

Craig A. Finseth

BASIC, as a programming language, offers a wide variety of features. There are, however, some features that most BASIC systems do not offer. Recursion and "on the fly" variable creation/deletion are two examples of these. In other languages, such as APL, these concepts are built-in as part of the language. Why is an APL system able to do these things while most BASIC systems cannot?

The prime reason for the difference is that APL stores its data in a way that is much more flexible, as well as making provisions for recursion. There is a price paid, in that the format for storing data can become rather complicated. This article is intended to make the concepts behind APL's data format visible. Don't expect to be presented with a "cut and dried" method for doing what would not be useful in itself. Instead, ideas and some "tools of the trade" will be evolved. In this way, you will be shown the reasons for a method of operation, as well as some other methods that do not work as well and the reasons why they don't.

It should be noted that this article is written about a specific application for these concepts. Because of this, there will be assertions made that something "must" be done a certain way. As with many things in the computer field, there are usually many other ways of doing something and some of them are probably better. Don't think that a certain way is the only way: think about *why* you are doing it that way and look for better methods.

Recursion

Recursion is the first difference mentioned, so let's start with it. First, what is recursion? A function is recursive when it calls itself. (It can also call a second function which calls the first, etc.) We shall use the classic factorial function example to illustrate the idea. (Note that this method is not necessarily the most efficient way of calculating factorials!)

Now, what are factorials? A factorial is the product of the integers from one to n. Thus, five factorial, written 5!, means 5*4*3*2*1. In the example we shall take advantage of a useful property of factorials in that n! is the same as n*(n-1)! (that is, 5! is the same as 5*4!). So, in our factorial function we can just have it call itself to figure out what (n-1)! is and then multiply that result by n. If you think that this could go on forever, you're right. Fortunately, mathematicians have conveniently defined zero factorial as one (0!=1). It's an easy matter to check to see if the argument is a zero; if so, we just return a one. One method of writing this function in a BASIC that allows multi-line function would be thus:

```
10 DEF FNF(N)
20 IF N<>0 THEN 50
30 LET FNF=1
40 GOTO 60
50 LET FNF=N*FNF(N-1)
60 FN END
```

This should look quite straightforward, with the exception of line 50 which is where the function calls itself. For this example, we are assuming that our BASIC system is able to handle recursive functions.

Suppose that we were to try to find 3!. Someplace in a program we might have a statement like the following:

```
666 LET X=FNF(3)
```

What would the execution of FNF look like? One way of showing what happens is to make a second copy of the function whenever it calls itself (this is also a possible method of implementing recursion). Figure 1 shows what it would look like.

In it, we inserted in each call to FNF a copy of the function. Along with this insertion, in each copy we replaced N by (N-1). (Remember that we want to do (n-1)!). Mentally setting N to 3, we can follow down the chain until we reach the end. (This diagram is tailored to the specific case of N=3. If N were to be ten, then there would be eleven copies of the function.) In the fourth copy, we see that (((3-1)-1)-1) is, indeed, zero and so we return a one. This one is multiplied oh ((3-1)-1) and the copy returns a one again. This result is multiplied by (3-1) which yields a two and that is finally multiplied by three, giving the final value of six. Three factorial is, of course, six.

Even though we can get the right answer this way, making copies of functions is not a very good way to operate. One reason is that you eat up both time and storage because you have to keep making copies of the function. Another reason is not to obvious. When we made copies of the function, we just substituted (N-1) for N whenever necessary. In our specific case, this worked fine, but how do we handle the case of the statement LET N=5? This would translate to LET (N-1)=5, which looks fishy, to say the least. If we were smart, we would change this to LET N=5+1, but this doesn't work on such other cases as LET(ABS(N))=5, not even to mention LET(A+B)=5. How do we translate that? Is N now equal to +5 or −5? It looks as if we need a new way of keeping track of what's happening, since it is not only inefficient, but nearly impossible to implement "blindly" as has been done here.

```
10 DEF FNF(N)
20 IF N <> 0 THEN 50
30 LET FNF=1
40 GOTO 60
50 LET FNF=N*  10 DEF FNF( (N-1) )
               20 IF (N-1) <> 0 THEN 50
               30 LET FNF=1
               40 GOTO 60
               50 LET FNF=(N-1)*  10 DEF FNF( ( (N-1)-1) )
                                  20 IF ( (N-1)-1) <> 0 THEN 50
                                  30 LET FNF=1
                                  40 GOTO 60
                                  50 LET FNF=( (N-1)-1)*  10 DEF FNF( ( ( (N-1)-1)-1) )
                                                          20 IF ( ( (N-1)-1)-1) <> 0 THEN 50
                                                          30 LET FNF=1
                                                          50 GOTO 60
                                                          50 LET FNF=( ( (N-1)-1)-1)*FNF(...)
                                                          60 FN END
                                  60 FN END
               60 FN END
60 FN END
```

Fig. 1. Using recursion to calculate factorials.

Stacks.

Let's digress for a moment and talk about food. In many cafeterias, there are spring-loaded platters in the counter that support stacks of plates, with only the top one visible. Suppose that we write a three on one plate and put it on top of the stack. We then take another plate, write a two on it, and put it on the pile. We add a plate with a one on it to the growing heap, and then finish off with a plate labeled zero. Now, when we remove the plates, the zero plate comes off first, followed by the one, then the two, and finally the number three plate. Note that the order in which we remove them is exactly reversed from that in which we placed them. Note also that there is only one number visible at any time, even though all of the numbers that are beneath it (placed on earlier) are "remembered." In our factorial function, we want N to take on successively the values 3, 2, 1, and 0 and then take on the same values in reverse order (1, 2, 3) as it returns from successive function calls. Going back to the plates, note that we placed plates on the stack, then removed only what we put on. There could have been plates underneath; if so, then they would not have been disturbed. This process is useful because our factorial function may not have been the only one called. For example, a function to compute sines may have been called, which then proceeded to call this one. This "stack of plates" concept, while it might not work, seems to give hope for finding a way to do recursion.

How do we write something in BASIC that can use this? There are two problems: interfacing to a stack of plates is not easy and it would be a very slow peripheral since we have to wait until someone comes to eat in order to see what is under the top plate. Besides, people might complain about numbers being written on their plates. Thus it seems reasonable to write a simulation of what is going on that will run using only BASIC and needing no fancy peripherals. What better object to use than an array, since we probably will wind up doing the same task over and over on many different values. We will start off by being completely arbitrary and do the following: DIM N(100), R(100). What, you ask, are we going to put in array N? N will contain the numbers that would otherwise have been written on the plates (3, 2, 1, etc.). Its use is to keep track both of what is in current use and the values to be remembered. R is the array that notes which line we were called from. It is being kept track of internally by BASIC itself and is being shown here to help you follow what is going on. (Remember that this BASIC can handle recursion.) Let's see what might be in N and R immediately after we call factorial for the first time:

N(1)=3 R(1)=666 (The function was called from line 666)

(Assume that any elements of the arrays N and R that are not specifically mentioned contain garbage.) We will now continue until FNF calls itself for the first time (the second call). Now, N and R look like this:

```
N(1)=2  R(1)=50
N(2)=3  R(2)=666
```

Until now, there has been no question about what the value of N is. Now, there are two entries in the array and there must be an agreed-upon method for figuring out which is the most current one. (Such an agreement is called a convention, and few, if any, conventions are industry-wide.) Judging by what we did to get into this situation, we shall adopt the convention, temporarily, that the currently active element is N(1). Thus, whenever we encounter an N in the function, we use the value in N(1) to tell us the current value for N. When each number is placed on top of the stack (this corresponds to making another copy of the function; this is called *pushing down* the stack), all of the other values can be moved one slot further down to "make room." As we shall see later, removing the top value (returning from a function, called *popping* the stack) allows all of the buried values to be moved one step towards the top.

We will continue executing the function and stop when N is zero and FNF has just been set to one (for the first time).

```
N(1)=0  R(1)=50
N(2)=1  R(2)=50
N(3)=2  R(3)=50
N(4)=3  R(4)=666
```

If you feel a bit lost now, go back and look at the expanded diagram (Fig. 1). We are currently on line 30 of the most deeply nested copy of the function. Let us continue on until we are just about to return from the copy of FNF where N=1 (line 50 of the next-most-deeply nested copy).

N(1)=1	R(1)=50
N(2)=2	R(2)=50
N(3)=3	R(3)=666

Note how the stack has popped. It should be apparent by now what will occur when we let execution continue until we return to line 666. Both of our problems have now been solved and the "stack of plates" works. If this array structure is incorporated within the BASIC system, then there no longer need be more than one copy of the function. We also can handle the statement LET N=5* because we can modify N(1) all we want, since it will be thrown away upon returning.

There still is a very inefficient process going on here. So far, each time we push something onto the stack, we laboriously move each entry down one place. This is very wasteful of time. This copying is done because we want the current-use position to be constant (the first element) but we can let the remembered values move about. The trouble is that there are usually more remembered values than current ones. So let's flip the stack end-over-end and fix the position of the remembered values, while allowing the position of the current one to change. We have to keep track of this position, so we shall introduce the variable P (for pointer, if you really must know) to do this.

The previous four snapshots will be reprinted in the new format:

N(1)=3	R(1)=666	becomes	N(1)=3	R(1)=666	P=1
N(1)=2	R(1)=50	becomes	N(1)=3	R(1)=666	P=2
N(2)=3	R(2)=666		N(2)=2	R(2)=50	
N(1)=0	R(1)=50	becomes	N(1)=3	R(1)=666	P=4
N(2)=1	R(2)=50		N(2)=2	R(2)=50	
N(3)=2	R(3)=50		N(3)=1	R(3)=50	
N(4)=3	R(4)=666		N(4)=0	R(4)=50	
N(1)=1	R(1)=50	becomes	N(1)=3	R(1)=666	P=3
N(2)=2	R(2)=50		N(2)=2	R(2)=50	
N(3)=3	R(3)=666		N(3)=1	R(3)=50	

In the first example, nothing much is accomplished. However, in going from the first to the second, no moving around was needed but only the incrementing of P and the placement of new values. In going from the third to the fourth example, instead of having to move up three entries in each array, all that happened was that P was set to P-1 (decremented). (Elements N(4) and R(4) don't even have to be erased. They just become part of that garbage that we mentioned earlier.) The only price for this increase in efficiency is that you must use N(P) instead of N(1) whenever you want to refer to the current value. Note that in neither case does it make a difference whether or not there was already something on the stack (P might just as well have been, say, ten to start).

The size, 100, which was picked arbitrarily, actually does have some significance. When you select the stack size, you must be sure to allocate enough room for the largest number of items to be remembered that you will ever come across. This can lead to having large amounts of empty space tied up in stacks that is not likely to ever be used. The common solution to this is to combine the various stacks into one large stack. For our example, we could do something like the following:

P=2	S(1)=666	the entry from R
	S(2)=3	the entry from N

To put the next function call on the stack, the 50 (from R) would go in S(3) (the odd element) and the 2 (from N) would go in S(4) (the even element). The main difference is that P would now go up by 2. The effect of all this is to have one array of size 200 instead of two arrays of size 100. Now, 2x100=200 and you might ask, where is the savings coming from? The space savings doesn't start until the things to be stored get more complicated. This savings comes when two or more stacks being combined do not always grow at the same rate. An example might serve to illustrate.

Function calls are the only things that we have been putting on the stack so far. There is another class of related things that also is convenient to put on the stack when writing an interpreter. This class is mainly concerned with evaluating expressions, and consists of open parentheses, subscript calculations, and operands for operators that have been delayed in execution (that is, in 4+5*6, the 4 must be held somewhere while the 5*6 is evaluated). Function calls can occur anytime and so can open parentheses, etc., but they occur independently and the occurrence of one does not imply the occurrence of the other (with the exception of the parentheses around the function's arguments). Usually, there are many more open parentheses, etc., than function calls.

Since we have two things which occur independently, we can make good use of the "common extra space." If there is a sudden rash of function calls, the common space can be used to handle it. The same storage, at a different time, can also be used for open parentheses, etc. Thus, we don't have to dedicate it to any one particular use. (The situation can arise where we want to call a lot of functions *and* evaluate a complicated statement. Here, the only solution is more storage.)

The entries on the stack containing the function calls or open parentheses will probably be of different, if not varying, lengths. An example of a varying-length entry is a function call, since it can have anywhere from zero arguments on up. Thus, it might prove useful to keep track of the lengths of the entries. In general, it somehow makes sense to place this length at the start of the entry (start being defined as the first part of the entry that you see). Later on, there will be a reason given for keeping those lengths around.

Believe it or not, you now know enough to implement recursive functions in your BASIC system, either by modifying the system itself or more simply by simulating. The main difficulty—in either case—is figuring out how to handle R (the return addresses), since it is that which most systems lack in a readily usable form. This is part of what APL can do that most BASICs do not.

Variable Creation/Deletion

Already we have seen how APL could do recursion, and that is the first of the differences mentioned. Variable creation/deletion "on the fly" is the other difference mentioned. We will start by examining what exactly it is that we are creating and deleting. Since we are talking about variables, it seems logical to start by keeping track of their names. APL, unlike BASIC, offers multi-character variable names, so it seems time to apply the length concept and store the number of characters in the name followed by the name itself. Another logical thing to do is to note down the value of the variable. Since APL allows multi-dimensional arrays, the number of dimensions (whether the name represents a simple variable, an array,

*This method allows us to do what we mean in a recursive program. When we have a function call itself (that is, FNF(N-1)) what we mean is that N in the new copy should have a value one less than that in the calling copy. This is accomplished by N(1)=N(2)-1 in the calling sequence. Now, modifying the current N will not affect the old values. A variable that has this property is said to be local to the function.

a matrix, etc.) should be stored, followed by the length of each dimension (another application of the length concept) followed by the value(s) itself (themselves). Alternatively, there could be "flag values" telling when, for example, the dimension list ends, eliminating the need for a number of dimensions entry. There are additional ways of doing this. Note that functions can be stored in a similar way, so the later discussions can be applied to either. Since what we have here amounts to a variable definition (as in definition of a variable), that makes as good a term as any to use when referring to the whole thing. The variable definition itself, of course, can have a length associated with it.

To keep track of all the variable definitions, we will store them in a table, simply listing them one after another. (Needless to say, this method will be replaced with a more efficient one!) Now, the table looks like this:

variable definition #1
variable definition #2
variable definition #3
•
•
•

To add a new variable definition to the list, you simply tack it on the end. To delete one, you could just replace it with zeros or some other marker. (This same marker could fill in the rest of the "scratch area" that this table is stored in.) There are three ways of modifying the definition that are of import here: making the new definition (1) shorter, (2) the same length, or (3) longer than the old one. If you make the new definition the same length, there is no problem. If you make it shorter, you can fill in the extra space with the marker value. If it is now longer, you can fill in the entire old definition with the marker and tack a new definition on the end.

This method is undesirable for three reasons. First, by always moving large chunks of data around, gaps tend to develop very quickly. Second, while this method allows recursion, it requires much too much shuffling of data to be efficient. One major inefficiency is that when figuring out which N to use, you might use the first one that you come across. This means that, when calling a function, if there is already an N in the table, you must insert a *new* definition such that the new definition is found first when searching the table. This entails pushing the old definition to the end of the table and replacing it with the new one—perhaps only temporarily. And then there is always modifying the new definition .... Last, this makes it nearly impossible to enforce order (alphabetic, for instance,) on the table.

The first problem can be solved by breaking definitions into smaller chuncks. Whenever a chunk has to be modified, only that part need generate a gap. This tends to keep the gap-size small, and, by not duplicating data unnecessarily, uses up storage at a slower rate.

The second problem is somewhat more complicated. It really gets emphasis when you are searching the table to find a name. To do this, you must find a name, then go element by element until you find a marker, then element by element until you find the next name. Needless to say, this is very slow. For an improvement, remember those lengths that we insist upon hanging on to without giving a reason? Instead of having to search through the whole table item-by-item, you can use these lengths to skip over the variable definitions entirely. Now, all that you have to do is search through marker values to find the start of the next name entry. And, if you were to hang on not only to the current lengths but also the original lengths (before shrinking), you could skip over the whole area and go directly from name to name. You now have a linked list on your hands (don't worry—it isn't contagious). Simply stated, a linked list is one in which you have a data block, with a pointer to the next data block, etc. (See Figure 2.)

This, however, is still not the best of all possible worlds. When definitions get deleted entirely, it can become very difficult to keep the table correct. This can be taken care of, in effect, by forgetting that the deleted variable was there at all. This can be done by causing the pointer which pointed to the variable-to-be-deleted-now to point to the next entry. Inserting an entry is just the opposite case. (See Figure 3.) Moving an entry is easy; you just cause the pointer to the entry to be moved now to point to the new location. (See Figure 4.)

This leads to the interesting prospect of the next name not having to be anywhere in particular. In other words, when you have to move a definition because it no longer fits in its present slot, all that has to change is the pointer in the preceding entry and, as far as the list is concerned, nothing has happened. This nicely solves the third problem in that it is now trivial to keep the list in alphabetical order (or any other order, for that matter). Also, recursion can be supported much more easily because the new definition can be added to the front of the

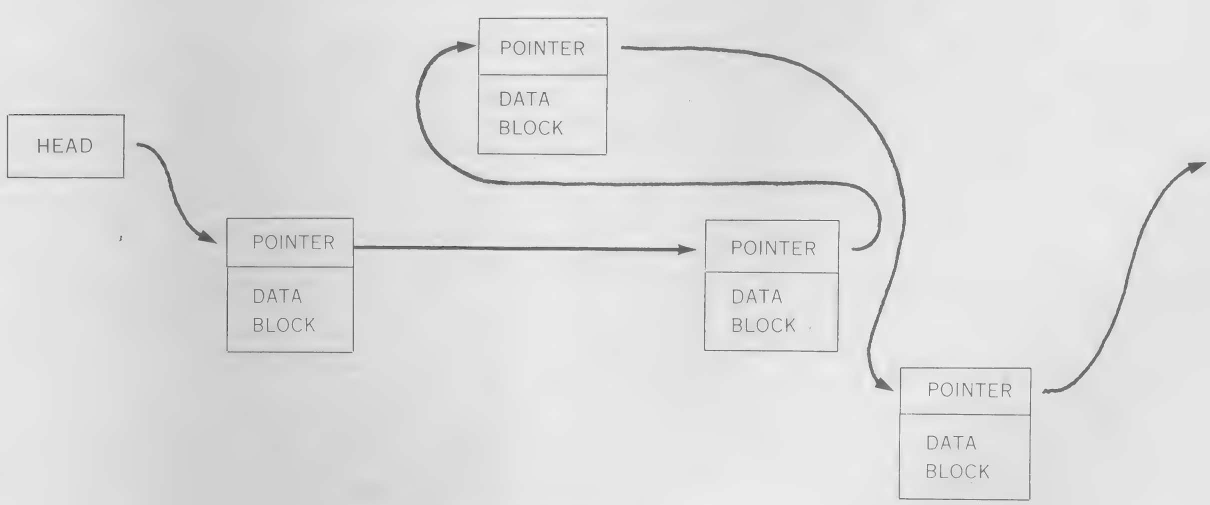


Fig. 2. Linked list.

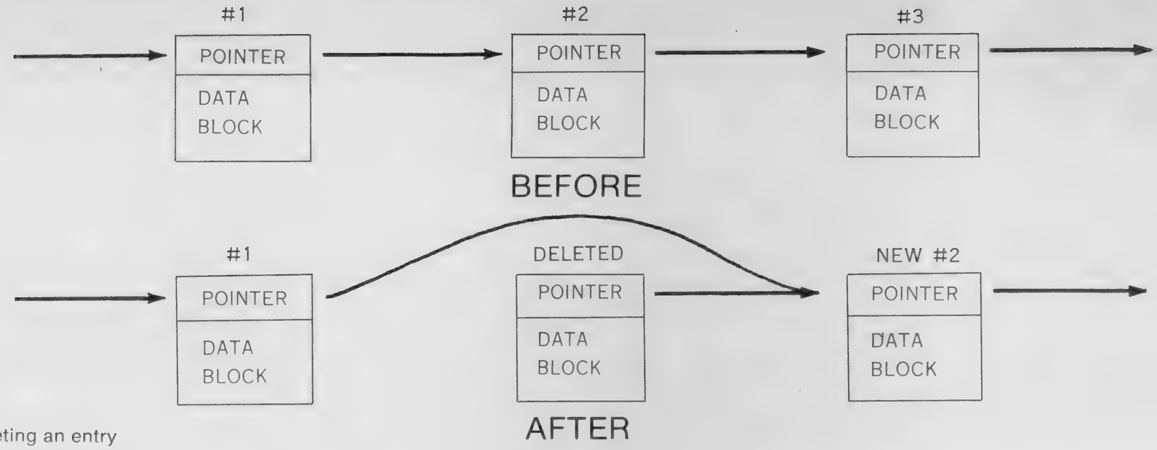


Fig. 3. Deleting an entry

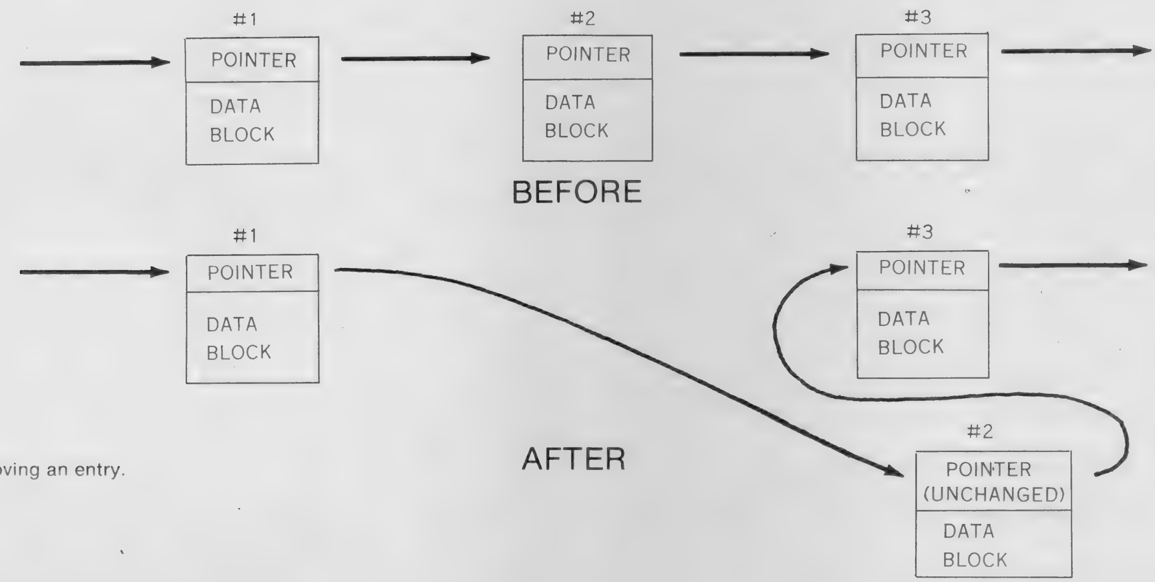


Fig. 4. Moving an entry.

list, even though it might be located at the physical end. There is still one minor flaw in this idea, and it is tied up in what the garbage-collect (described later) does. The salient point is that the garbage-collect moves stuff around, and the next entry is relative to where you are now. Thus, there are two ways for the pointers to get messed up. The most obvious change is to use absolute (relative to some fixed point) locations instead of relative (to where you are now) locations. Now, the pointers only have one way to get messed up. (Besides, by using absolute locations, each entry becomes independent of where it is located and can be moved around more easily.)

Although at first glance there doesn't seem to be any way of stopping the lists from going on forever, there's a convention regarding this which is to use an illegal or unused address as the final pointer (often zero). By checking for this address, you can find out when to stop.

It should begin to be apparent that when storing information like pointers, lengths, and other such things, we are using storage for non-data items. This is part of the trade-off between low overhead (amount of space used for "housekeeping" purposes) and flexibility in the system. This means that if you only want the data, then you are very limited in what your options are, while if you want to keep a "description" of the data, you have more options and even—to some extent—the ability to ask questions. In a simple example, you can write a program to average ten numbers, or you can write one to average N numbers. In the latter case, you have overhead in that you have to store N. You also have an option in what you can do and you can even ask what N is. The same concept applies on a much larger scale when you start keeping as many extras as we are. There is a different trade-off: that between low overhead and low execution time. The extra information requires space, but it significantly cuts down on the time needed to execute a program.

Garbage-Collect

Finally, there is yet another toll exacted when programs become dynamic (constantly changing). Remember all those gaps that are created whenever you update a definition (or almost anything, for that matter)? These accumulate until a sizeable fraction of storage can be eaten up by them. They must then be packed down from time to time to keep the program from running out of space while there is still unused storage sitting around. It is this packing down that causes all of the lists to be moved around. The program that does this is called a garbage-collect and must—unfortunately—be individually written for each application. Garbage-collects are another use for all of those lengths, since it is convenient for the garbage-collect to know how long a chunk of data is in order to move it. Although this requires extra space to store, it saves large amounts of time and even cuts down on the size of the garbage-collect routine. It should be noted that there are ways of avoiding a garbage-collect. These

involve having fixed-length chunks of data (to eliminate small gaps) and keeping a linked list of unused space. This method is often used in managing disk space, since garbage-collects on disc are very time-consuming.

Name Table

Turning back to our table of variable definitions, we find that it now could contain only the names of the variables. In the entry for each name might be a pointer telling us where the rest of the definition is. The rest of the definition may even be sitting right next to the name, but that doesn't matter. The reason for this is that when we are searching the table, we are only interested in the names until we find the correct one. Only then do we look at the rest of the definition. Hence, from now on, we will call it the name table.

The name table, along with the stack containing information on function calls, etc., can tell us everything that is going on in and around the program. This, then, is called the environment and, even though the details or even the major parts can change from program to program, the term environment still refers to the "general picture" of what is going on.

Our name table is a linked list which has the names in some order. The only ordering that we will assume is that when a function is called, its parameters' (arguments') definitions will be placed in the front of the list so that we will find them before their earlier definitions (either in the main program, another function, or another recursion level), if there were any. When a function finishes (returns), the most recently added definitions (that is, those that it added— isn't it nice how stacks work?) are removed. Thus, parameter definitions only exist after a function has been called and before it finishes. Thus they are local to the function and are called local variables. (In almost all languages that support recursion, there is a way to define a variable as local without its having been passed as a parameter.) The way then to determine which definition of a variable to use (assuming that there is more than one definition) is to start at the beginning and use the first occurrence of the name.

This method is reasonably straightforward, but it does have one drawback. If you are calling a function that has five local variables and this function calls itself ten times, then there are fifty entries in the table followed by whatever was there before. There will probably be two types of variables that you are looking for: local variables (the most recent five of the fifty) or anything else (what is after the fifty). It seems rather inefficient to have to look at forty-five names without a hope that what you want will be there. Now, the situation on the stack will change whenever we call another function or return from one. This happens a lot less often, on the average, than trying to find the definition of a variable. We can, therefore, afford to spend a little extra time doing some polishing up of the name table at that time. The polishing up, of course, consists of effectively removing the entries of those forty-five "extra" definitions.

When we put a function call on the stack, we also could put a listing of the local variables for that function with the call. This is useful in telling us which variables to delete from the name table when we return. Suppose, for the moment, that we would also store a pointer to the definition for the variable that it replaces (or, say, a zero if there wasn't one). This eliminates the need for the old entry in the table and it can be deleted. We now have only the new entry in the table and, in the function call, a pointer to the old entry. (See Figure 5. The function F has local variables A and C. There is only one entry for A in the name table, yet all the information can be recovered.) Thus, there is no information lost and upon returning, it is a simple task to restore the table by replacing the pointers to the new entries with those of the old and deleting all names that have our marker (zero) in their function call entry.

This concludes the introduction to some ideas, concepts, and "tools of the trade." Of course, many details were omitted, but then these vary greatly depending upon the exact nature of what your program is to do. You should now be able to see how APL—or any languages that are dynamic and/or allow recursion—put these techniques together to come up with a working system. ■

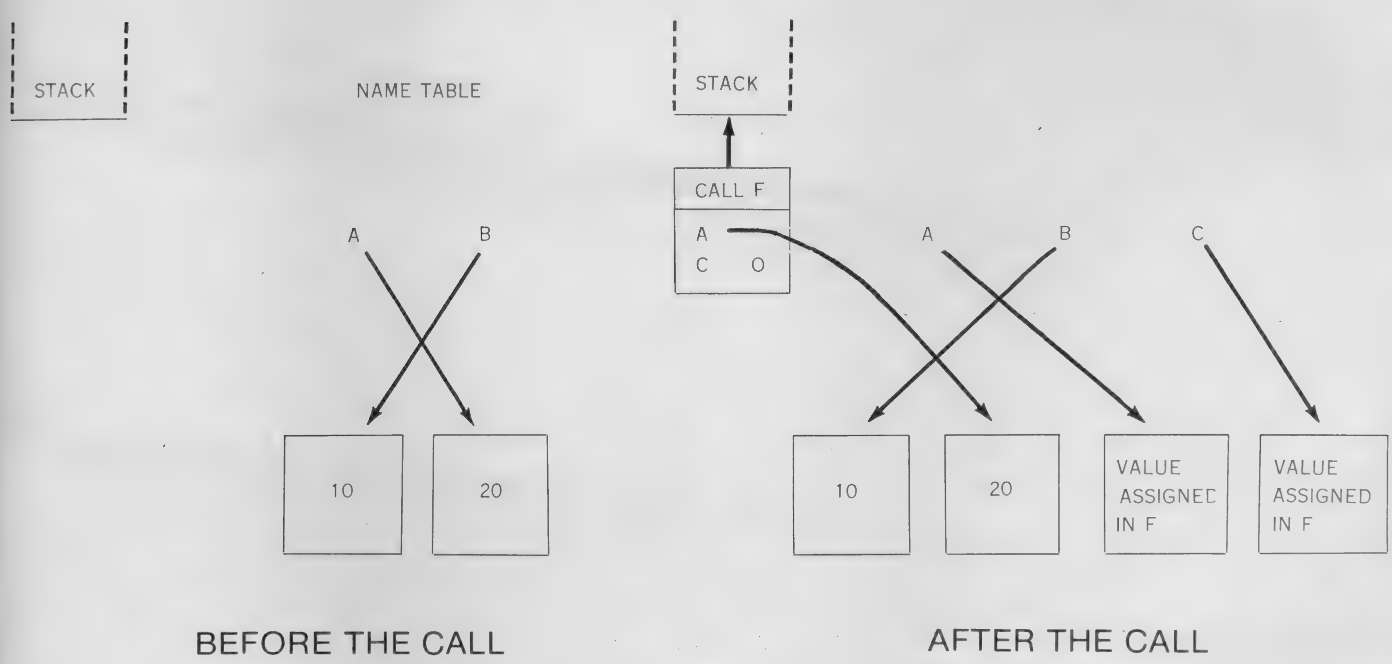
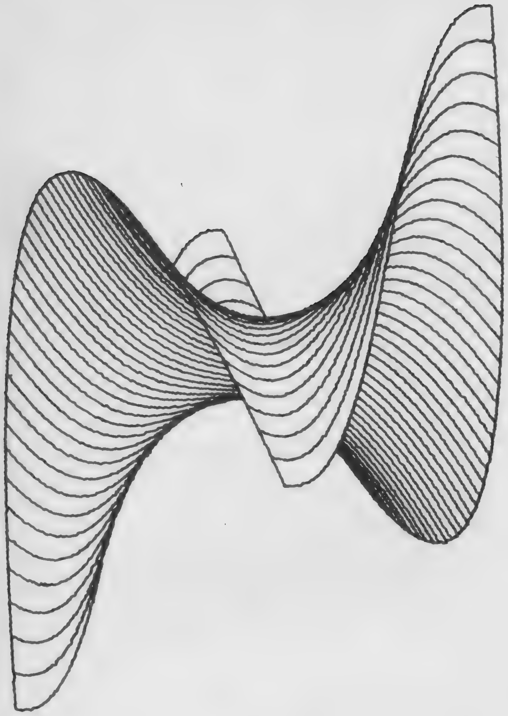
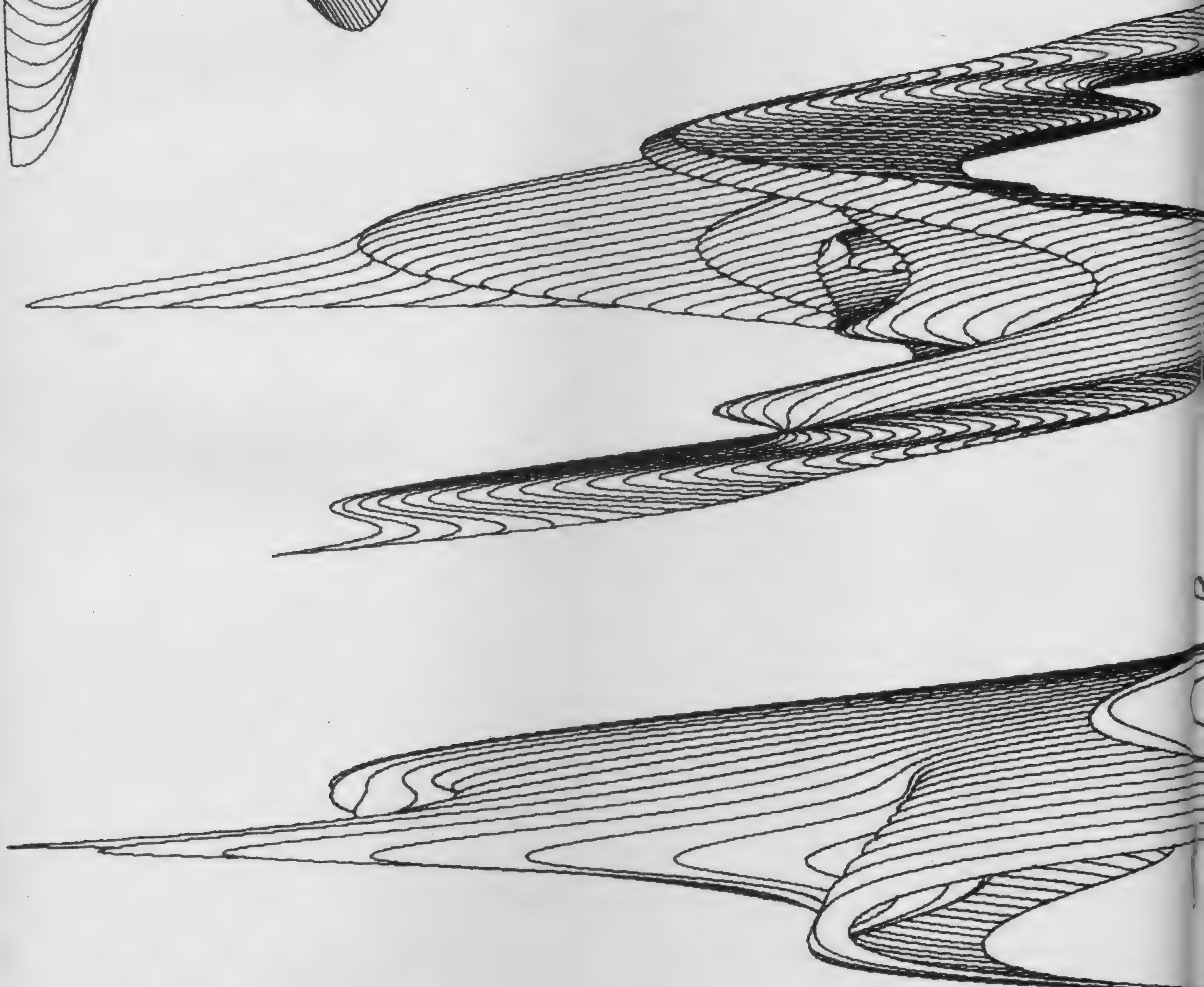


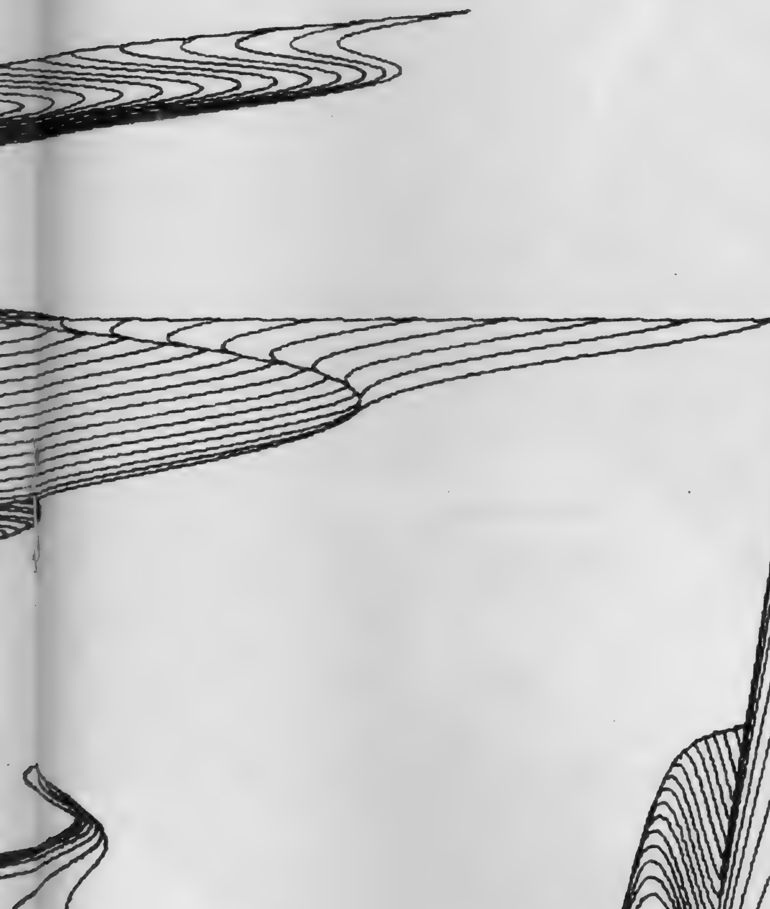
Fig. 5. Stack and the name table.

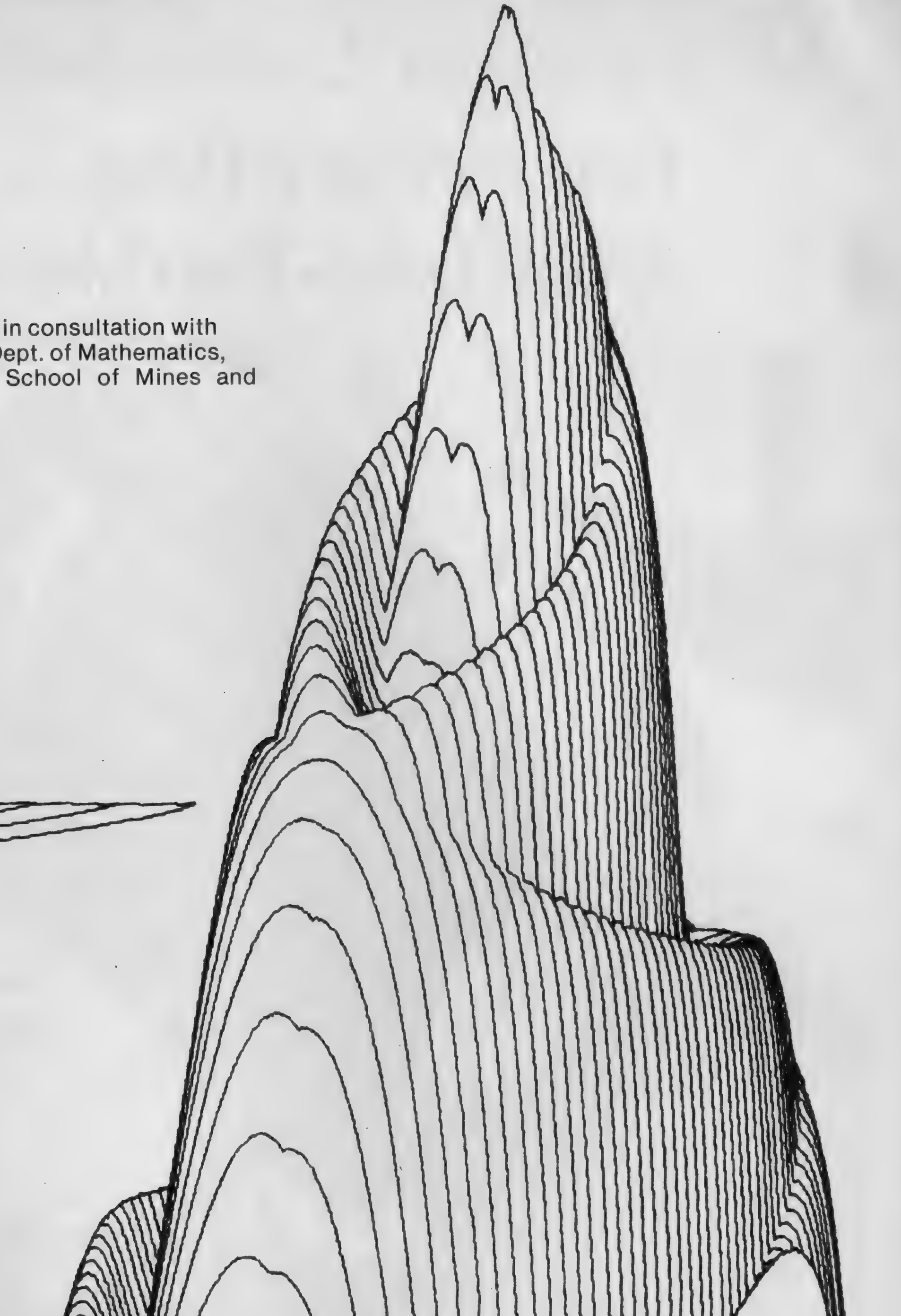
PLOTTER ART





By Nick Mykris in consultation with David Ballew, Dept. of Mathematics, South Dakota School of Mines and Technology.





CAI: Further Considerations for Presenting Multiple-Problem Types

Part three in this five-part series on Computer-Assisted Instruction looks at incorporating a sliding grade level into programs that generate several types of problems.

Laura L. McLaughlin

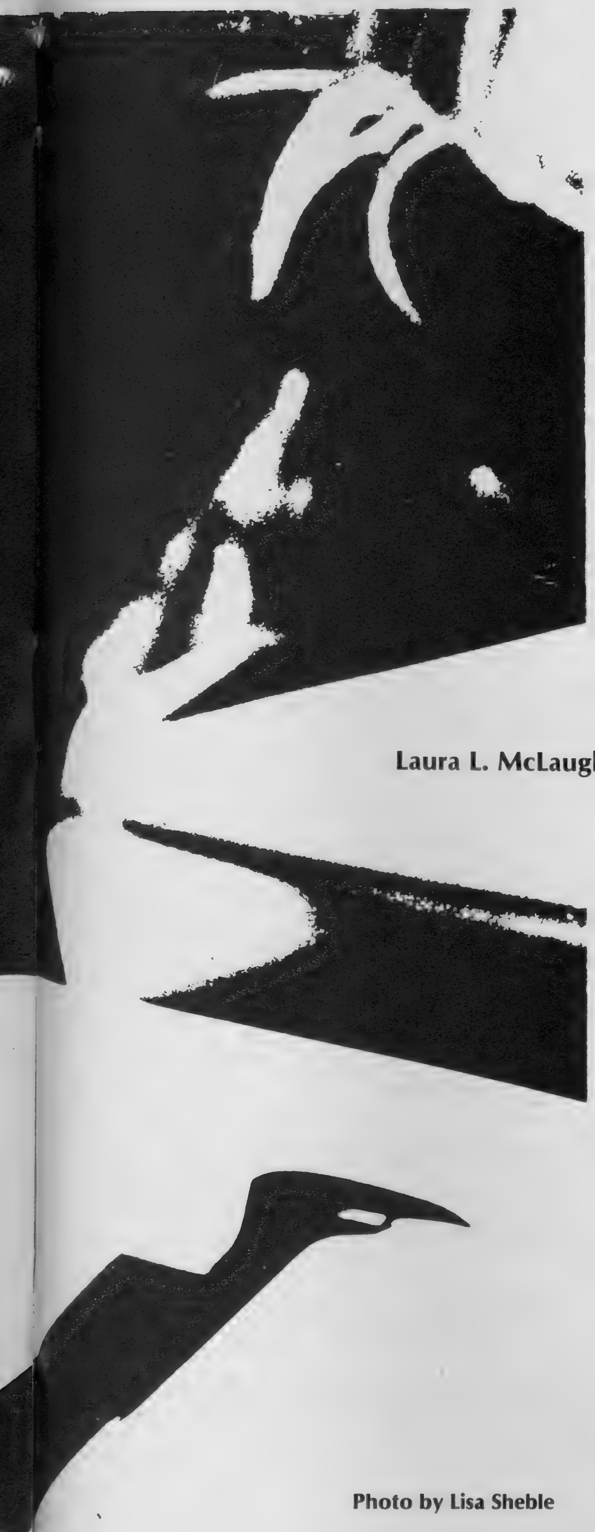


Photo by Lisa Sheble

```
B>RUN24 MATHART
BASIC-E INTERPRETER: VER K1.3

HI! WHAT'S YOUR NAME? TRACY
I GUESS WE'VE NEVER MET BEFORE TRACY
WHAT GRADE ARE YOU IN? 2

OKAY, LET'S PLAY WITH SOME NUMBERS

   1
 + 4
------
? 5

THAT'S RIGHT, TRACY

LET'S TRY ANOTHER...

  21
 + 15
------
? 36

THAT'S RIGHT, TRACY

LET'S TRY ANOTHER...

  41
 + 15
------
? 50

WRONG, TRY AGAIN? 55

WRONG, TRY AGAIN? 56

THAT'S RIGHT, TRACY

LET'S TRY ANOTHER...

  24
 + 14
------
? 38

THAT'S RIGHT, TRACY

LET'S TRY ANOTHER...

  54
 + 12
------
? 64

WRONG, TRY AGAIN? 54

WRONG, TRY AGAIN? 62

YOU DON'T SEEM TO UNDERSTAND THIS ONE TRACY
THE CORRECT ANSWER IS 66

LET'S TRY ANOTHER...

41 + 7 =
```

EXAMPLE 1

```
10      DEF FNIN$(X)=LEFT$(STR$(X+.5),INT(LOG(X)/2.302585+1))
        DIM G(6)
        NAMES$="NAME.RND"
        NAMES=1
        FILE NAMES$(128)
        IF END #NAMES THEN 300
        INPUT "HI! WHAT'S YOUR NAME";NAME$
        RANDOMIZE
100     COUNT=COUNT+1
        READ #NAMES; NAMEF$, G(1), G(2), G(3), G(4), G(5), G(6)
        IF NAME$=NAMEF$ THEN 500
        GOTO 100
300     PRINT "I GUESS WE'VE NEVER MET BEFORE ";NAME$
        INPUT "WHAT GRADE ARE YOU IN";G(1)
        G(1)=G(1)+.3
        FOR I=1 TO 5
        G(I+1) = G(I)
        NEXT I
500     PRINT
        PRINT "OKAY, LET'S PLAY WITH SOME NUMBERS"


7000    PRINT
        PRINT "THAT'S ENOUGH FOR NOW ";NAME$
        PRINT "COME BACK AGAIN SOON"
        PRINT #NAMES, COUNT; NAME$, G(1), G(2), G(3), G(4), G(5), G(6)
9999    END
A>
```

The previous article in this series discussed a method of calculating a sliding grade level based on the range of the numbers presented in a particular problem. But how can we incorporate this concept into a program that handles multiple-problem types and what other factors should we consider?

Let's say we want a program that will generate six types of problems for grades 1-4: addition, subtraction, and multiplication in both the vertical and horizontal formats. We want this program to keep track of a student's grade level for each type of problem separately. We can then give him practice problems, in each area, that fall within a half-grade on either side of his current level for that particular type of problem. The program should constantly adjust the student's grade levels based on the accuracy of his responses.

We would also like the program to be both flexible and concise. Flexibility is important so that we can later expand and/or modify the types of problems to be presented or the grade levels to be covered. The need for conciseness is evident to all (except maybe those with their own personal IBM 370). To accomplish this, we are going to make extensive use of the BASIC FOR/NEXT and GOSUB verbs.

But before we can generate any problems, we must have a means of obtaining the student's grade levels. Since we are now talking about six different levels based on problem type and format, it is no longer practical to expect the student to enter his grade level data. Therefore, we are going to build a file with one record for each student. It will contain his name and a level for each type of problem. Example 1 shows the beginning and end of our program.

(1) At 100, we search through the file for the student's name. When a match is found we can proceed with the session, since we have his current levels.

(2) If there is no record for a student, we must ask him what grade he is in. This will provide us with initial values (see Line 300).

(3) Since his true level is probably somewhere between what he answers and the next higher grade, we will set our initial values up by three-tenths. Remember, the program will adjust these numbers in either direction based on his ability, so the initial grade levels are not absolute.

(4) The end of the program (at 7000) either updates or creates his record with the levels that have been adjusted during this session.

Just a word about the function (FNIN$) defined at the beginning of the program. As you may know, each of the different BASICs available has its own "quirks" (or, more accurately, problems). This program was developed using BASIC-E Version K1.4. One of its "quirks" is that on occasion it will incorrectly convert a numeric integer into a string variable as a non-integer number (for example, 93 becomes "93.00001"). The function FNIN$ is used to correct this problem. When writing a CAI program, it is essential that all such inaccuracies be identified and corrected or it will result in confusing, rather than helping, the student. Check your BASIC, nothing is perfect.

Now, what kinds of common subroutines can we use? Well, Example 2 shows four of the more obvious ones. The routine at 4000 will set up to three random numbers based on the value of G1 (the student's grade level). The formula for determining the maximum value of any one of the numbers is the same as used in the previous article for addition problems (Addend = 1.73 x Grade Level^4.)* However, we have provided the formula with greater flexibility by making the values for both FACTOR and POWER variable. This way they may be changed based on problem type. Note that the value of NUM must have been previously set for both this routine and the two that follow. Routines for printing a problem vertically or horizontally are shown at 5000 and 5500 respectively, with the type of problem determined by the value of OP$. The last routine, at 6000, will get the student's answer and process it. Three incorrect responses are accepted prior to showing the right answer. Note that this could be easily changed to vary depending on grade level, if we later determined that allowing only two mistakes would be more appropriate for the younger student. We then decide if the problem should effect his grade level and do the calculations if necessary.

We are now ready to write that part of the program which will actually generate the problems. The code in Example 3 will generate addition (1000), subtraction (2000), and multiplication (3000) problems. Let's look at it a little more closely and see what it really does:

(1) The values for G1 (student's grade level), OP$ (type of problem), FACTOR and POWER (parameters for calculating variables) are set based on the type of problem.

(2) Five problems of each type are presented in succession (with the exception that no multiplications are generated if his grade level for horizontal multiplication is below 2.5). By presenting the same math concept multiple times in succession, while at the same time limiting the number, we will give him the opportunity to learn from his mistakes but decrease the possibility of boredom or

EXAMPLE 2

```
4000    G2=G1-.5+RND
        MAX=INT(FACTOR*G2^POWER)
        A=INT(RND*MAX)
        B=INT(RND*MAX)
        IF NUM=2 THEN C=0:RETURN
        C=INT(RND*MAX)
        RETURN


5000    PRINT
        PRINT "   ";
        IF A=0 THEN PRINT A ELSE PRINT FNIN$(A)
        IF NUM=2THEN 5100
        PRINT "   ";
        IF C=0 THEN PRINT C ELSE PRINT FNIN$(C)
5100    PRINT " ";OP$;" ";
        IF B=0 THEN PRINT B ELSE PRINT FNIN$(B)
        PRINT "------"
        RETURN


5500    PRINT
        IF A=0 THEN PRINT A; ELSE PRINT FNIN$(A);
        PRINT " ";OP$;" ";
        IF NUM = 2 THEN 5600
        IF C=0 THEN PRINT C; ELSE PRINT FNIN$(C);
        PRINT " ";OP$;" ";
5600    IF B=0 THEN PRINT B ELSE PRINT FNIN$(B);
        PRINT " = "
        RETURN


6000    INPUT ANSWER
        IF ANSWER = R THEN 6200
        W=W+1
        IF W>2 THEN 6100
        PRINT
        PRINT "WRONG, TRY AGAIN";
        GOTO 6000
6100    PRINT
        PRINT "YOU DON'T SEEM TO UNDERSTAND THIS ONE ";NAME$
        PRINT "THE CORRECT ANSWER IS ";
        IF R=0 THEN PRINT R ELSE PRINT FNIN$(R)
        IF G2>G1 THEN GOTO 6300 ELSE GOTO 6250
6200    PRINT
        PRINT "THAT'S RIGHT, ";NAME$
        IF G2<G1 THEN 6300
6250    MAX=SQR(MAX/FACTOR)
        IF POWER = 4 THEN MAX=SQR(MAX)
        G1=.9*G1+.1*MAX
6300    W=0
        IF OP$="-" AND G(5)<2.5 AND I=10 THEN RETURN
        IF OP$="X" AND I=10 THEN RETURN
        PRINT
        PRINT "LET'S TRY ANOTHER..."
        RETURN

A>
```

*Some terminals use this "upside-down saucer" to indicate exponentiation, rather than an up-arrow.

EXAMPLE 3

```
1000    G1 = G(1)
        OP$ = "+"
        FACTOR = 1.73:POWER = 4
        FOR I = 1 TO 10
        IF G1<4 THEN NUM=2 ELSE NUM=3
        GOSUB 4000
        IF G2<2.5 AND I<6 THEN GOSUB 4500
        IF G2<3 AND I>5 THEN GOSUB 4500
        R=A+B+C
        IF I<6 THEN GOSUB 5000 ELSE GOSUB 5500
        GOSUB 6000
        IF I=5 THEN G(1)=G1:G1=G(2)
        NEXT I
        G(2)=G1
        GOSUB 6500


2000    G1=G(3)
        OP$="-":NUM=2
        FOR I=1 TO 10
        GOSUB 4000
        IF A<B THEN X=A:A=B:B=X
        IF G2<3 AND I<6 THEN GOSUB 4500
        IF G2<3.5 AND I>5 THEN GOSUB 4500
        R=A-B
        IF I<6 THEN GOSUB 5000 ELSE GOSUB 5500
        GOSUB 6000
        IF I=5 THEN G(3)=G1:G1=G(4)
        NEXT I
        G(4)=G1
        GOSUB 6500


3000    G1=G(5)
        IF G1<2.5 THEN 7000
        OP$="X"
        FACTOR=.68:POWER=2
        FOR I=1 TO 10
        GOSUB 4000
        R=A*B
        IF I<6 THEN GOSUB 5000 ELSE GOSUB 5500
        GOSUB 6000
        IF I=5 THEN G(5)=G1:G1=G(6)
        NEXT I
        G(6)=G1
        GOTO 7000

A>
```

frustration.

(3) The value of NUM (number of variables) is set for addition based on the grade level of the problem (G2), while for subtraction and multiplication it is set to 2.

(4) When done with a particular problem type, the student's grade level for that type is updated.

We could have put the generation of all three problem types into one large FOR-NEXT loop. This approach would have saved us close to 50% of the code necessary to produce the problems, but would have made modifications and/or additions much more difficult (the old ease of maintenance/size & efficiency trade-off). The choice of coding technique should be based upon the extent of modification envisioned to achieve the final iteration.

The addition and subtraction loops also call two routines we have not yet defined. Take a look at Example 4. The routine at 4500 is called from addition and subtraction only for problems of that below a particular level. It will insure that addition problems will not require a "carry" if both numbers are greater than 9, and that subtractions will not need a "borrow." This kind of special editing (independent of the range of the variables) must be considered for each type of math concept presented. Otherwise, we run the risk of completely frustrating the student by expecting him to attempt something which could well be far beyond his capabilities. Similarly, since we do not want to give multiplication too soon, the routine at 6500 is used to set the student's grade level to a value that will allow the generation of multiplication problems only after he has reached a particular level (grade 3) in one of the other types.

Now that we have a basis from which to work, it will be relatively easy to expand upon. Subroutines can be written for generating different formats (mixed operations, fractions) or more operations (division, square roots). Also, additional checks can be included to give a student a larger variety of problems and eliminate those which he has already mastered.

In the next article, we will discuss some of the considerations that should be made concerning the interaction between the student and the computer. Careful thought must be given to such things as edit requirements and presentation formats so that the student does not need to learn a whole new set of rules. Remember, for CAI to be a useful tool, it must be something that is easy and comfortable to use. ■

EXAMPLE 4

```
4500    IF A<10 OR B<10 THEN RETURN
        TSTA$=RIGHT$(FNIN$(A),1)
        TSTB$=RIGHT$(FNIN$(B),1)
        TSTA=VAL(TSTA$)
        TSTB=VAL(TSTB$)
        IF OP$="-" THEN 4600
        IF TSTA + TSTB < 10 THEN RETURN
        IF TSTA > 4 THEN A=A-5
        IF TSTB > 5 THEN B=B-5
        RETURN
4600    IF TSTA - TSTB >= 0 THEN RETURN
        IF TSTA < 5 THEN A=A+5
        IF TSTB > 5 THEN B=B-5
        RETURN


6500    IF G(5) >= 3 THEN RETURN
        IF G1<3 THEN RETURN
        G(5)=3
        G(6)=3
        RETURN

A>
```

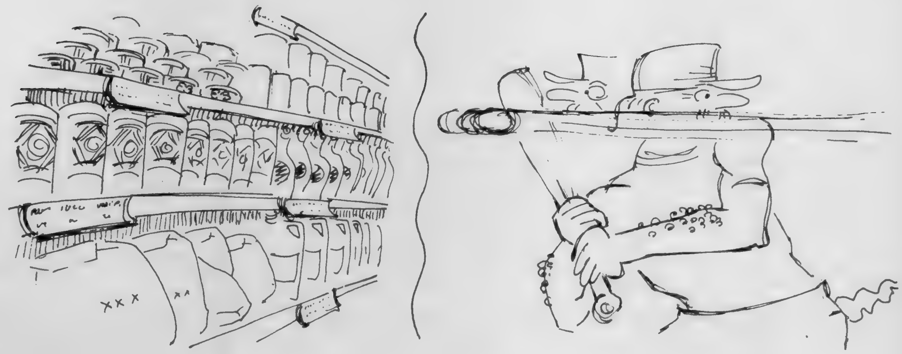
Part III in this series examines READ...DATA, RESTORE, INT, ON...GOTO and the fantastic RND.

THE 8-HOUR WONDER

All About BASIC Programming in One Long Day (or Eight Short Nights)

Thomas A. Dwyer

2.5 HOUR: SHELF LABELS AND BATTING AVERAGES



The word is out. You're the first one on your block with a computer, and the calls are starting to roll in. First the butcher, then the baker, and now—the local sports writer. Seems he needs to crank out a list of batting averages fast, his calculator is broken, and he never did understand long division. Meanwhile, the corner grocer wonders if you could maybe print him unit-price tags of the kind used in supermarkets. Is there a simple way to handle both requests?

READ...DATA

One way to kill several birds with one stone in the world of computing is to realize that different programs may have similar structures, differing mainly in the data they use. For this reason, it would be nice if the data could be kept more or less separate from the program itself. This also makes it easier to expand or revise data later on.

Here's how this idea works in BASIC for the batting average problem: →

```
LIST

10 PRINT "PLAYER #","AT BAT","HITS","BAT.AVG."
20 READ N, B, H
30 PRINT N, B, H, H/B
40 GO TO 20
50 DATA 1, 50, 19
60 DATA 2, 43, 10
70 DATA 3, 51, 13
80 DATA 4, 49, 17
90 END


RUN

PLAYER #        AT BAT          HITS            BAT.AVG.
 1               50              19              .38
 2               43              10              .232558
 3               51              13              .254902
 4               49              17              .346939
OUT OF DATA at line 20
```

WHEN ROUNDED TO THREE DECIMAL PLACES, THIS RATIO GIVES WHAT BASEBALL FANS CALL A 'BATTING AVERAGE'.

When this program reaches line 20, it is told to READ enough data to load the variables N, B, and H. So it looks for a DATA statement (which it finds at line 50), and "uses up" the first three pieces of data it finds. You can think of what happens as follows:

20 READ N, B, H

50 DATA 1, 50, 19

You should also picture this data as having been "used up":

20 DATA ~~1~~, ~~50~~, ~~19~~

The program next prints a line of output (line 30), and then does a "GO TO 20". This means it *again* reads data, but starting with the first "fresh" (unused) piece of data it can find. In our example, this is found at line 60, so the second time around our loop we have:

20 READ N, B, H

60 DATA 2, 43, 10

This process continues until no more "fresh" data can be found, at which time an "out of data" message is printed.

IMPORTANT: The data can be distributed over DATA statements any way you wish, provided it is in the *order* expected by the READ statement. For example, lines 50, 60, 70, and 80 could also be written as two statements:

50 DATA 1, 50, 19, 2, 43, 10

60 DATA 3, 51, 13, 4, 49, 17

or even as one statement:

50 DATA 1, 50, 19, 2, 43, 10, 3, 51, 13, 4, 49, 17

Actually, a program always treats all data as one big list. The READ statement simply goes down the list, "eating up" the data in "gulps." In our example, each "gulp" consists of three numbers, and it's *up to you* to make sure the groups of three correspond to N, B, and H.

Here's a similar program for our grocer friend. All we have to do is change our interpretation of what the variables mean, and use data appropriate to grocery prices. We'll also print things a little differently so the grocer can actually cut up the output to make shelf labels. →

```
LIST

5 PRINT "-------------------------------------------------------------------
10 READ N, Q, P
20 PRINT "PRODUCT #","QTY.IN OZ.","PRICE","UNIT PRICE"
30 PRINT N, Q, P, 100*P/Q; "CENTS PER OZ."
40 GO TO 5
50 DATA 1, 15, 1.29, 2, 4, .69, 3, 32, 2.49
60 END


RUN

-------------------------------------------------------------------
PRODUCT #     QTY.IN OZ.    PRICE         UNIT PRICE
 1             15            1.29          8.6 CENTS PER OZ.
-------------------------------------------------------------------
PRODUCT #     QTY.IN OZ.    PRICE         UNIT PRICE
 2             4             .69           17.25 CENTS PER OZ.
-------------------------------------------------------------------
PRODUCT #     QTY.IN OZ.    PRICE         UNIT PRICE
 3             32            2.49          7.78125 CENTS PER OZ.
-------------------------------------------------------------------
OUT OF DATA at line 10
```

Improving These Programs

One of the nice things about writing programs is that once the basic idea is up and running, it's easy to add improvements. For example, both of the above programs suggest several kinds of additions. We'll describe five of these, and illustrate the last three.

(a) Limit the number of decimals to what people expect: .367 instead of .366666 for a batting average, 13.5¢ instead of 13.49999 for a unit price. There are two ways to do this. One uses the INT function which will be explained in Section 2.7 [in the Jan/Feb 1978 issue]. The other uses PRINT USING, explained in Chapter 3.*

(b) It would be nice to have words or names printed instead of product or player numbers. The best way to do this is to use string variables, explained in Chapter 4.*

(c) It would be convenient to allow grocery data to be given in both pounds and ounces. This is easy to do. Here's one way:

20 READ N, L, Z, P

25 LET Q = 16*L + Z

This means 15 oz.

50 DATA 1, 2, 7, 1.31, 2, 0, 15, .89

We agree that this means product #1 contains 2 lbs 7 oz and costs $1.31. Statement 25 then converts Q to 39 oz.

*Ed. Note: For this chapter you'll have to read the author's book, "An Amateur's Guide to Personal Computing," to be published by Addison Wesley.

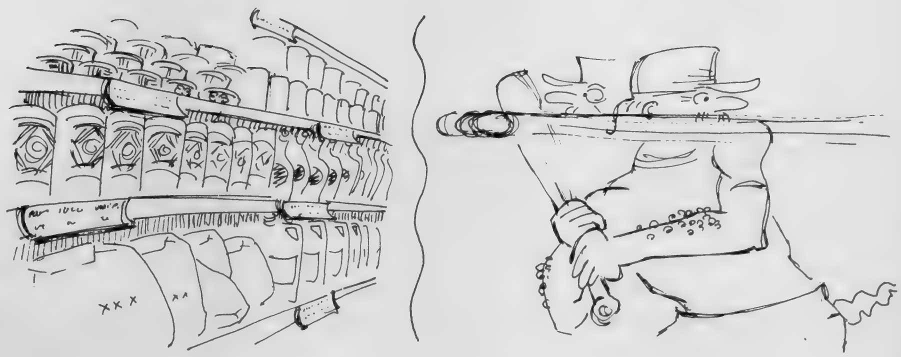
Part III in this series examines READ...DATA, RESTORE, INT, ON...GOTO and the fantastic RND.

THE 8-HOUR WONDER

All About BASIC Programming in One Long Day (or Eight Short Nights)

Thomas A. Dwyer

2.5 HOUR: SHELF LABELS AND BATTING AVERAGES



The word is out. You're the first one on your block with a computer, and the calls are starting to roll in. First the butcher, then the baker, and now—the local sports writer. Seems he needs to crank out a list of batting averages fast, his calculator is broken, and he never did understand long division. Meanwhile, the corner grocer wonders if you could maybe print him unit-price tags of the kind used in supermarkets. Is there a simple way to handle both requests?

READ...DATA

One way to kill several birds with one stone in the world of computing is to realize that different programs may have similar structures, differing mainly in the data they use. For this reason, it would be nice if the data could be kept more or less separate from the program itself. This also makes it easier to expand or revise data later on.

Here's how this idea works in BASIC for the batting average problem:

```
LIST

10 PRINT "PLAYER #","AT BAT","HITS","BAT.AVG."
20 READ N, B, H
30 PRINT N, B, H, H/B
40 GO TO 20
50 DATA 1, 50, 19
60 DATA 2, 43, 10
70 DATA 3, 51, 13
80 DATA 4, 49, 17
90 END

RUN

PLAYER #      AT BAT        HITS          BAT.AVG.
 1             50            19            .38
 2             43            10            .232558
 3             51            13            .254902
 4             49            17            .346939
OUT OF DATA at line 20
```

WHEN ROUNDED TO THREE DECIMAL PLACES, THIS RATIO GIVES WHAT BASEBALL FANS CALL A 'BATTING AVERAGE'.

When this program reaches line 20, it is told to READ enough data to load the variables N, B, and H. So it looks for a DATA statement (which it finds at line 50), and "uses up" the first three pieces of data it finds. You can think of what happens as follows:

20 READ N, B, H

50 DATA 1, 50, 19

You should also picture this data as having been "used up":

20 DATA ~~1~~, ~~50~~, ~~19~~

The program next prints a line of output (line 30), and then does a "GO TO 20". This means it *again* reads data, but starting with the first "fresh" (unused) piece of data it can find. In our example, this is found at line 60, so the second time around our loop we have:

20 READ N, B, H

60 DATA 2, 43, 10

This process continues until no more "fresh" data can be found, at which time an "out of data" message is printed.

IMPORTANT: The data can be distributed over DATA statements any way you wish, provided it is in the *order* expected by the READ statement. For example, lines 50, 60, 70, and 80 could also be written as two statements:

50 DATA 1, 50, 19, 2, 43, 10

60 DATA 3, 51, 13, 4, 49, 17

or even as one statement:

50 DATA 1, 50, 19, 2, 43, 10, 3, 51, 13, 4, 49, 17

Actually, a program always treats all data as one big list. The READ statement simply goes down the list, "eating up" the data in "gulps." In our example, each "gulp" consists of three numbers, and it's *up to you* to make sure the groups of three correspond to N, B, and H.

Here's a similar program for our grocer friend. All we have to do is change our interpretation of what the variables mean, and use data appropriate to grocery prices. We'll also print things a little differently so the grocer can actually cut up the output to make shelf labels.

```
LIST

5 PRINT "-----------------------------------------------------------------
10 READ N, Q, P
20 PRINT "PRODUCT #","QTY.IN OZ.","PRICE","UNIT PRICE"
30 PRINT N, Q, P, 100*P/Q; "CENTS PER OZ."
40 GO TO 5
50 DATA 1, 15, 1.29, 2, 4, .69, 3, 32, 2.49
60 END


RUN

-----------------------------------------------------------------
PRODUCT #      QTY.IN OZ.     PRICE          UNIT PRICE
 1              15             1.29           8.6 CENTS PER OZ.
-----------------------------------------------------------------
PRODUCT #      QTY.IN OZ.     PRICE          UNIT PRICE
 2              4              .69            17.25 CENTS PER OZ.
-----------------------------------------------------------------
PRODUCT #      QTY.IN OZ.     PRICE          UNIT PRICE
 3              32             2.49           7.78125 CENTS PER OZ.
-----------------------------------------------------------------
OUT OF DATA at line 10
```

Improving These Programs

One of the nice things about writing programs is that once the basic idea is up and running, it's easy to add improvements. For example, both of the above programs suggest several kinds of additions. We'll describe five of these, and illustrate the last three.

(a) Limit the number of decimals to what people expect: .367 instead of .366666 for a batting average, 13.5¢ instead of 13.49999 for a unit price. There are two ways to do this. One uses the INT function which will be explained in Section 2.7 [in the Jan/Feb 1978 issue]. The other uses PRINT USING, explained in Chapter 3.*

(b) It would be nice to have words or names printed instead of product or player numbers. The best way to do this is to use string variables, explained in Chapter 4.*

(c) It would be convenient to allow grocery data to be given in both pounds and ounces. This is easy to do. Here's one way:

20 READ N, L, Z, P

25 LET Q = 16*L + Z

This means 15 oz.

50 DATA 1, 2, 7, 1.31, 2, 0, 15, .89

We agree that this means product #1 contains 2 lbs 7 oz and costs $1.31. Statement 25 then converts Q to 39 oz.

*Ed. Note: For this chapter you'll have to read the author's book, "An Amateur's Guide to Personal Computing," to be published by Addison Wesley.

(d) The "out of data" message terminates the program. But suppose we want the program to continue and do other things? How do we handle this? (Read on!)

(e) We may also want a program to re-use data that's been "scratched out." How do we "restore" such used-up data?

Here's a revision of the batting average program that answers both of these questions:

```
LIST

10 PRINT "PLAYER #  CLASS   AT BAT   HITS   BAT.AVG."
20 READ N, C, B, H
30 IF N = 0 THEN 110
40 PRINT N; TAB(10);
50 IF C = 0 THEN 90
60 PRINT  "VETERAN";
70 PRINT TAB(18);B;TAB(27);H;TAB(34);H/B
80 GO TO 20
90 PRINT "ROOKIE";
100 GO TO 70
110 RESTORE
115 PRINT
120 PRINT "BAR GRAPH OF PLAYER BATTING AVERAGES"
130 READ N, C, B, H
140 IF N = 0 THEN 220
150 PRINT "PLAYER #";N;
160 FOR K = 1 TO 100*(H/B+.005)
170 PRINT "*";
180 NEXT K
190 PRINT
200 GO TO 130
210 DATA 1,0,50,12,2,1,49,18,3,1,51,17,4,0,43,15,0,0,0,0
220 END
```

THIS EXPRESSION CONVERTS AN AVERAGE OF .24 TO 24 ASTERISKS, .367347 TO 37 ASTERISKS, ETC.

```
RUN

PLAYER #  CLASS    AT BAT   HITS   BAT.AVG.
 1        ROOKIE    50       12     .24
 2        VETERAN   49       18     .367347
 3        VETERAN   51       17     .333333
 4        ROOKIE    43       15     .348837

BAR GRAPH OF PLAYER BATTING AVERAGES
PLAYER # 1 ************************
PLAYER # 2 *************************************
PLAYER # 3 *********************************
PLAYER # 4 ***********************************
```

We used the second data item in each group of four as a "code," with 0 meaning rookie and 1 meaning veteran. Line 50 tests C to see what this code is, and then branches to the appropriate PRINT statement.

RESTORE

The revised batting average program uses the DATA in line 210 twice. The first time it's used to produce a table of batting averages. This is done in lines 10 to 100. This part of the program keeps looping back to line 20 to get new data. But the fifth time this happens, it finds the "phony" data 0, 0, 0, 0. We agree that storing a zero in N signals the end of data. The signal is picked up in line 30 which then causes a branch to the second part of our program (the bar-graph routine from lines 120 to 200).

IMPORTANT: Even though we only need one zero in N to signal end of data, it is essential that four zeros be put at the end of the data statement. This is because the READ statement has four variables to fill, and will squawk with an error message if it doesn't find four data items.

Now you can see what the special statement 110 RESTORE does. The first part of the problem "uses up" all the data. (What happens is that a "pointer" moves along the data to keep track, and when the pointer gets to the end of the list, the program knows it's "out of data.") The RESTORE statement *resets* this pointer back to the first data item. Now all the data can be used again. (Of course, re-running a program also resets the pointer, but that doesn't help in our example because we would never reach the bar-graph part.)

SELF-TEST

1. Simulate running this program:

```
10 LET T1 = 0
20 LET T2 = 0
30 READ A, B
40 IF A = 0 THEN 90
50 PRINT A, B, A/B
60 LET T1 = T1 + A
70 LET T2 = T2 + B
80 GO TO 30
90 PRINT "TOTALS AND OVERALL RATIO"
100 PRINT T1, T2, T1/T2
120 DATA 6, 4, 10, 5, 4, 1
130 DATA 0,0
140 END
```

2. Write and run a program to help balance your check book. It should be like Problem 3 at the end of Section 2.2 [Jul/Aug issue] but use READ and DATA instead of INPUT.

3. Write and run a student record program that has a DATA statement for each student in a class as follows:

```
100 DATA 101, 16, 75, 80, 65, 90
```

Student #	Age	Quiz grades

The program should print out a class roster with the grade average of each student. Finally it should print the average age in the class, the average grade for each quiz, and the overall class average.

2.6 HOUR 6: COMPUTER GAMES OF CHANCE

"What a pity this isn't a sin!" Those are supposed to be the words of the novelist Stendahl upon tasting ice cream for the first time. They sound more like the utterance of a computer-center director trying to find a rationale for evicting the game-playing devotees who clutter up his system.

But personal computers are a different story, and the wages of gaming on your own system are an intellectual refreshment that comes in more flavors than found in all the ice-cream stands ever franchised.

This section explains the features of BASIC that help make this endless variety possible. We'll start by first answering one of the questions we raised in the last section: How do you make a number like

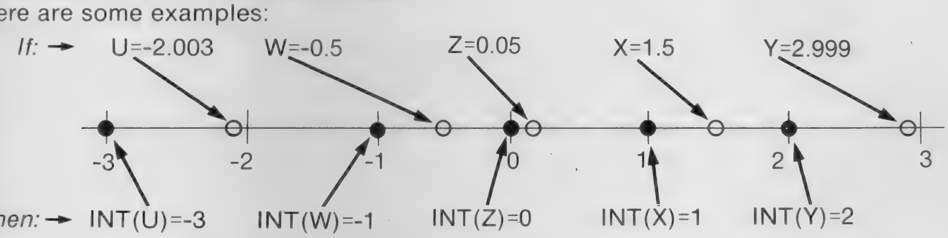
.343687 print as .344?
or .264689 print as 26.46?
or .891246 print as 89?

INT

One way to control the number of decimal places in a number is to use the INT function of BASIC. If a statement says

10 LET Y = INT(X)

the INT(X) part means that X is to be first "processed" by something called the INT (integer) function. What comes out of the processing is an integer just to the *left* of X on the number scale. Here are some examples:



The following program shows some more examples of the differences between X and INT(X):

```
LIST

5 PRINT "X", "INT(X)", "X/3", "INT(100*X/3)"
10 FOR X = -2 TO 2 STEP .5
20 PRINT X, INT(X), X/3, INT(100*X/3)
30 NEXT X
40 END


RUN

X              INT(X)         X/3            INT(100*X/3)
-2             -2             -.666667       -67
-1.5           -2             -.5            -50
-1             -1             -.333333       -34
-.5            -1             -.166667       -17
 0              0              0              0
 .5             0              .166667        16
 1              1              .333333        33
 1.5            1              .5             50
 2              2              .666667        66
```

To use INT for getting an answer in dollars and cents with only two decimal places (remember the UNIT PRICE program?) we can use the expression INT(100*X)/100. That's because

if	X = 1.36782
then	100*X = 136.782
and	INT(100*X) = 136
so	INT(100*X)/100 = 1.36

To change a batting average to three decimal places we can use a similar trick:

if	A = .367891
then	1000*A = 367.891
and	INT(1000*A) = 367
so	INT(1000*A)/1000 = .367

One more thing. To round this answer "up" in the third decimal place, use INT(1000*A + .5)/100 = .368

> SELF-TEST
> 1. Modify and test run the BATTING AVERAGE and UNIT PRICE programs using the above techniques to appropriately change the number of decimal places in the output.

Meanwhile, back at the Casino

RND

One feature no computer language should be without is a random-number generator. This is a built-in routine that produces a "surprise" number each time it's used. When a statement like

10 LET X = RND (0)

is executed, a number between 0 and 1 is produced "randomly," and stored in X. Here's a simple test program you can use to see what these numbers look like in your BASIC.

```
LIST

10 PRINT "RANDOM NOS. WITH VARIOUS MULTIPLIERS"
20 FOR K = 1 TO 10
30 LET X = RND(0)
40 PRINT X, 10*X, 100*X, INT(100*X)
50 NEXT K
60 END


RUN

RANDOM NOS. WITH VARIOUS MULTIPLIERS
 .771027        7.71027        77.1027        77
 .78183         7.8183         78.183         78
 .75174         7.5174         75.174         75
 .473969        4.73969        47.3969        47
 .781555E-1     .781555        7.81555        7
 .203217        2.03217        20.3217        20
 .5159          5.159          51.59          51
 .266449        2.66449        26.6449        26
 .955597        9.55597        95.5597        95
 .335541        3.35541        33.5541        33
```

THIS STRANGE NUMBER IS .0781555 IN DISGUISE (SEE SECTION 2.8).

NOTE: Your version of BASIC will probably produce a different sequence of random numbers, but the general idea is the same. Also, RND (0) may have to be changed to RND(1) in some BASICs. (In standard BASIC the argument—the number in parenthesis—is ignored, but in other versions it's got to be as specified in the user manual.)

Each time RND is used in line 30, it's as though a new number X from a secret list is revealed. The word "random" means that no one number will appear more frequently than any other. To say it another way, if you generate a lot of random numbers, they should be distributed equally over the interval used (zero to one is the interval in the example above). A program that can be used to check out the distribution of numbers produced by the random number generator in your BASIC is suggested in the first SELF-TEST question at the end of this section.

Constants in BASIC

The numbers used in BASIC programs are called constants. So far we have used.

(1) *Integer* constants like -3, 4, 27893, and

(2) *Floating-point* (or real) constants like .0831, 3.1416, and -896.28.

Another way to write a floating point constant is shown in the first column of the preceding test program where the number .781555E-1 appears.

This is called the "exponential" or "scientific" notation for writing constants. What .781555E-1 really means is

$.781555 * 10^{-1}$

But 10^{-1} means $1/10^{1}$ (remember, $10^{1} = 10$), so this number is really (.781555) * (1/10) = .0781555. Similarly,

.781555E-2 = $.781555 * 10^{-2}$ = .00781555, and
.781555E-3 = $.781555 * 10^{-3}$ = .000781555, and so on.

SIMPLE RULE #1

E-3 means "move the decimal point 3 places to the *left.*"

.781555E-3 = .000781555

Scientific notation is used to save space when representing very small and very large numbers. (You can see that .781555E-10 takes less room to print than .0000000000781555.)

A similar notation is used to represent large numbers. For example,

.8965E+1 means $.8965 * 10^{1}$ = .8965 * 10 = 8.965,
.8965E+2 means $.8965 * 10^{2}$ = .8965 * 100 = 89.65,
.8965E+ means $.8965 * 10^{3}$ = .8965 * 1000 = 896.5,

and so on. Here, the space saving shows up for very large numbers.

For example,

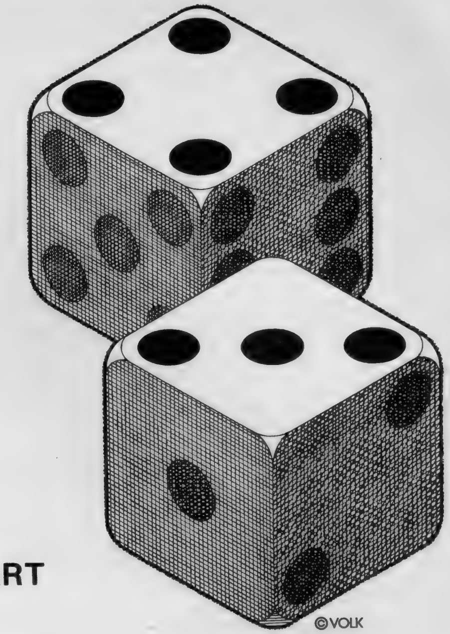
.8965E+18 = 896500000000000000.

SIMPLE RULE #2

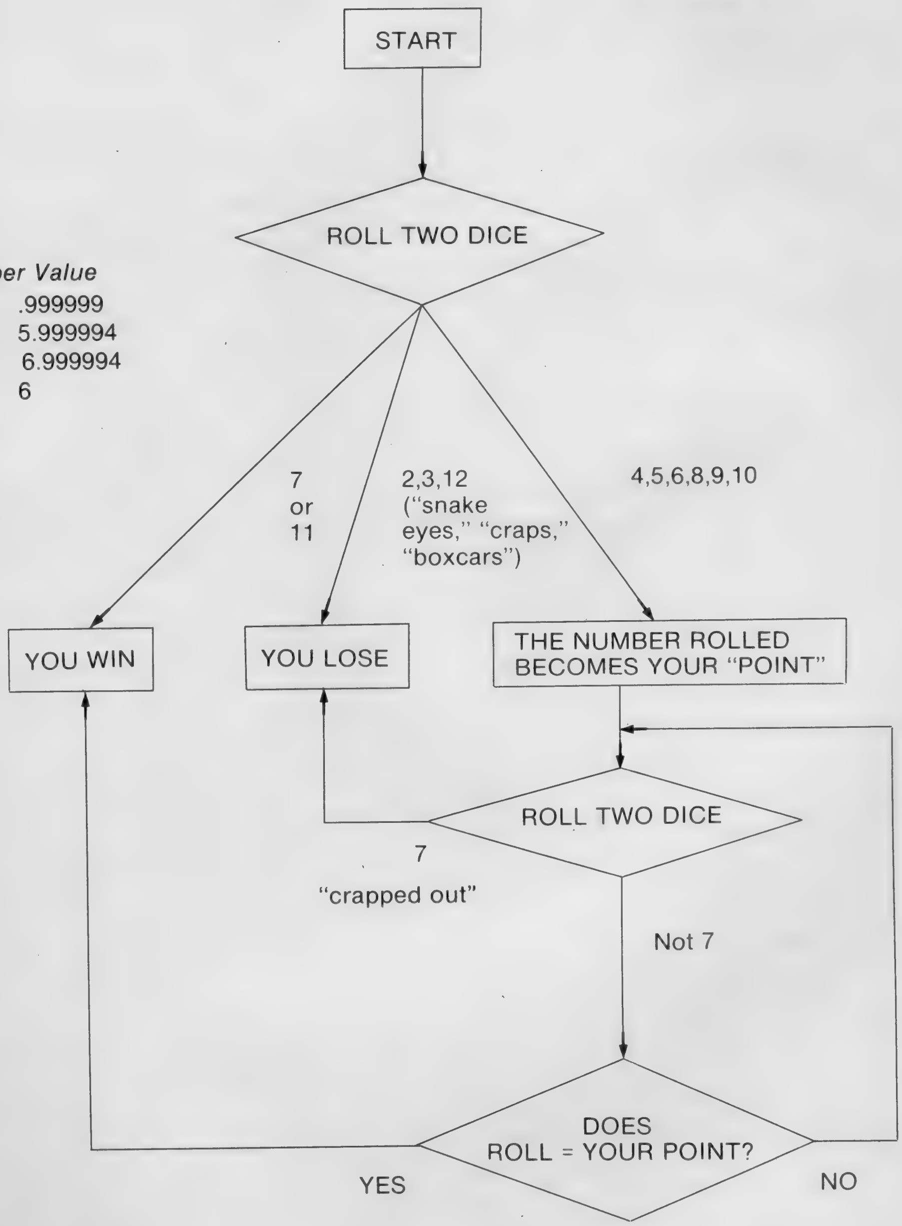
E+18 means "move the decimal point 18 places to the *right*."

Simulated Craps

Now let's get back to the use of RND by writing a program to play craps. The usual rules for this dice game can be summarized in flowchart form as follows:



CRAPS FLOW CHART



To write a program that simulates playing this game, we'll need two statements that simulate the roll of two dice by producing random integers from 1 to 6. The statements

```
30 LET D1 = INT(6*RND(0) + 1)
40 LET D2 = INT(6*RND(0) + 1)
```

do this because RND(0) produces numbers from 0 (zero) up to (but not including) 1. So for six decimal places we'd have:

		Lower Value		*Upper Value*
RND(0)	produces	.000000	to	.999999
6*RND(0)	produces	.000000	to	5.999994
6*RND(0)+1	produces	1.000000	to	6.999994
INT(6*RND(0)+1)	produces	1	to	6

Thus both D1 and D2 produce integers from 1 to 6. Mathematicians say this by writing that 1 D1 6 and 1 D2 6.

Note for Statistics Buffs: Tossing two dice with six sides gives numbers with a total value from 2 to 12. But you will *not* get the same effect by using a "super die" with eleven sides as follows:

```
10 LET D = INT(11*RND(0) + 2)
```

It's true that this statement will produce random integers from 2 to 12, but they will not show up with the same distribution you get from adding the results of tossing two six-sided dice. For example, with one "super die", the number 7 will show up 1/11 of the time. But with two regular dice, the number 7 can be formed in six different ways, each of which shows up 1/36 of the time. So on the average, a 7 will show up 6*(1/36) = 1/16 of the time, not 1/11.

General Formula for Transforming RND

As just shown, the formula

INT(6*RND(0) + 1)

transforms the random numbers so that they fall in the interval $1 \leq X \leq 6$. To generate random numbers in the range $A \leq X \leq B$ use the formula

INT((B-A+1) * RND(0) + A)

Examples: To generate integers from 50 to 85 use:

20 LET X = INT(36*RND(0) + 50)

To generate two-place decimals from .50 to .85 use:

20 LET X = INT(36*RND(0) + 50)/100

To generate integers from -90 to +80 use:

20 LET X = INT(171*RND(0) - 90)

Returning to the CRAPS program, here's a listing followed by a sample run:

```
RUN

SIMULATED CRAPS GAME--YOU START WITH $10
HOW MUCH DO YOU WANT TO BET? 2
ROLL IS 10
YOUR POINT IS 10
NEXT ROLL IS 8
NEXT ROLL IS 3
NEXT ROLL IS 6
NEXT ROLL IS 9
NEXT ROLL IS 6
NEXT ROLL IS 2
NEXT ROLL IS 7
TOUGH--YOU LOSE.  YOU NOW HAVE $ 8
WANT TO PLAY AGAIN (1=YES)? 1
HOW MUCH DO YOU WANT TO BET? 4
ROLL IS 8
YOUR POINT IS 8
NEXT ROLL IS 5
NEXT ROLL IS 7
TOUGH--YOU LOSE.  YOU NOW HAVE $ 4
WANT TO PLAY AGAIN (1=YES)? 1
HOW MUCH DO YOU WANT TO BET? 8
ROLL IS 3
TOUGH--YOU LOSE.  YOU NOW HAVE $-4
WANT TO PLAY AGAIN (1=YES)? 0
YOU ENDED WITH $-4 WON'T YOU EVER LEARN?
STOP at line 280
```

This version of the program has been written to make each statement as simple as possible. Questions 5 and 6 of the Self-Test section coming up make some suggestions for shortening the program.

Here's a run of the craps program. Your program may give different dice rolls because it has a different random-number generator.

```
LIST

5 RANDOMIZE
10 PRINT "SIMULATED CRAPS GAME--YOU START WITH $10"
20 LET D = 10                    D WILL KEEP TRACK OF DOLLARS YOU HAVE.
30 PRINT "HOW MUCH DO YOU WANT TO BET";
40 INPUT B
50 LET D1 = INT(6*RND(0) + 1)    FIRST ROLL OF DICE.
60 LET D2 = INT(6*RND(0) + 1)
70 LET R1 = D1 + D2
75 PRINT "ROLL IS";R1
80 IF R1=7 THEN 200
90 IF R1 = 11 THEN 200
100 IF R1 = 2 THEN 170
110 IF R1 = 3 THEN 170           NEXT ROLL OF DICE.
120 IF R1 = 12 THEN 170
130 PRINT "YOUR POINT IS";R1
140 LET R2 = INT(6*RND(0) + 1) + INT(6*RND(0) + 1)
145 PRINT "NEXT ROLL IS";R2
147 IF R2=7 THEN 170             YOU LOST, SO YOUR BET IS SUBTRACTED FROM D.
150 IF R2 = R1 THEN 200
160 GOTO 140
170 LET D = D - B
180 PRINT "TOUGH--YOU LOSE.  YOU NOW HAVE $";D
190 GOTO 220
200 LET D = D + B                YOU WON, SO YOUR BET IS ADDED TO D.
210 PRINT "YOU WIN!  YOU NOW HAVE $";D
220 PRINT "WANT TO PLAY AGAIN (1=YES)";
230 INPUT A
240 IF A = 1 THEN 30
250 PRINT "YOU ENDED WITH $";D;
260 IF D>10 THEN 290
270 PRINT "WON'T YOU EVER LEARN?"
280 STOP
290 PRINT "TALK ABOUT LUCK!"
300 END
```

RANDOMIZE

If you run the craps simulation program several times, you'll find that the rolls of the dice are the same for each run. This is because RND(0) always starts with the same "seed" value, and procudes each new number with the same algorithm. This is very helpful for debugging programs.

To make the numbers really surprise you, there is a feature in most versions of BASIC that creates a new seed number for each run. All you have to do to get this feature is to start your program with the statement

5 RANDOMIZE

To see what happens, run the craps program twice with RANDOMIZE, and twice without.

NOTE: Some versions of BASIC don't have a RANDOMIZE. Their normal way of operating is to give you a different sequence of random numbers on each run. For these systems, if you want the same sequence of random numbers on each run, you must put a statement like

5 Z=RND (-1)

at the beginning of the program. (Confusing? Agreed!)

ON... GOTO...

This is sometimes called the "computed GOTO" statement. It branches to different line numbers, depending on the value of a variable placed right after the word ON. Here's a program that demonstrates how it works:

```
LIST

10 PRINT "QUIZ:  WHO WAS THE 4TH MARX BROTHER?"
20 PRINT "1 = ZIPPO, 2 = HARRY, 3 = ZEPPO"
30 INPUT A
40 ON A GO TO 50, 70, 90
50 PRINT "NO, YOU'RE THINKING OF A CIGAR LIGHTER--TRY AGAIN."
60 GOTO 30
70 PRINT "YOU MAY BE WILD ABOUT HARRY, BUT THAT'S NOT RIGHT."
71 PRINT "TRY AGAIN."
80 GOTO 30
90 PRINT "BY GEORGE YOU'VE GOT IT!!"
100 END

Ready

RUN

QUIZ:  WHO WAS THE 4TH MARX BROTHER?
1 = ZIPPO, 2 = HARRY, 3 = ZEPPO
? 2
YOU MAY BE WILD ABOUT HARRY, BUT THAT'S NOT RIGHT.
TRY AGAIN.
? 1
NO, YOU'RE THINKING OF A CIGAR LIGHTER--TRY AGAIN.
? 3
BY GEORGE YOU'VE GOT IT!!
```

IF A=1, GO TO 50
IF A=2, GO TO 70
IF A=3, GO TO 90

Here's a program that uses RND with ON...GOTO... to generate random messages. If you analyze the output, you can see that RND must have produced the integers 4, 4, 4, 2, 1, 1, 3, 2, 4, 2 which caused branches to lines 100, 100, 100, 60, 40, 40, 80, 60, 100, 60

```
LIST

5 RANDOMIZE
10 FOR N=1 TO 10
20 LET K=INT (4*RND(0)+1)
30 ON K GO TO 40, 60, 80, 100
40 PRINT "HEE-";
50 GO TO 110
60 PRINT "HA-";
70 GO TO 110
80 PRINT "HIC-";
90 GO TO 110
100 PRINT "HO-";
110 NEXT N
120 END

RUN

HO-HO-HO-HA-HEE-HEE-HIC-HA-HO-HA-
```

SELF-TEST

1. Simulate running the following program, using a die to produce the random numbers in line 80. What application do you see for this program?

```
10 LET K1 = 0
20 LET K2 = 0
30 LET K3 = 0
40 LET K4 = 0
50 LET K5 = 0
60 LET K6 = 0
70 FOR N = 1 TO 600
80 LET R = INT(6*RND(0) + 1)
90 ON R GO TO 140, 150, 160, 170, 180, 190
140 LET K1 = K1 + 1
145 GO TO 210
150 LET K2 = K2 + 1
155 GO TO 210
160 LET K3 = K3 + 1
165 GO TO 210
170 LET K4 = K4 + 1
175 GO TO 210
180 LET K5 = K5 + 1
185 GO TO 210
190 LET K6 = K6 + 1
210 NEXT N
230 PRINT K1; K2; K3; K4; K5; K6
240 END
```

2. You didn't really do #1 completely did you? Six hundred die tosses is a bit much. To get some real insight about RND from this program, you should run it on your computer.

3. Play the CRAPS program using the strategy of doubling your bet each time. Will this always guarantee that you eventually come out ahead? What feature can be added to the program to make this strategy less threatening to the "house"?

4. Modify the CRAPS program so a FOR...NEXT loop controls how often it plays. Then run it for a large number of plays (say 100, 200, 300, etc.) printing only the final value of D. The program itself should make the bets, using various strategies (always bet $1, for example). See what you can discover about the odds of winning this game for various strategies.

5. Can you find five statements in the CRAPS program that can be replaced with a single statement?

Hint, try:

```
80 ON R1 GOTO 130, 170, 170, 130, 130, 130, 200, 130, 130, 130, 200, 170
```

6. Can you replace statements 50, 60, 70 and 75 in the CRAPS program with a single statement? ■

This fifth article in the series shows how indirect reasoning can lead to a contradiction and thus to the truth.

Thinking Strategies with the Computer: Contradiction

Donald T. Piele and Larry E. Wood

Let craft, ambition, spite
Be quenched in Reason's night,
Till weakness turn to might
Till what is dark be light
Till what is wrong be right!
Lewis Carroll

One of the first and possibly most elegant applications of the method of contradiction was given by Euclid in the 3rd century B.C. He proved that among the natural numbers 1, 2, 3, 4, ... there exists an infinite number of primes. (A prime number is one which has no factors other than itself and 1; for example, 2, 3, 5, 7 are prime, while 9 is not.) Euclid's idea is delightfully simple and illustrates the problem-solving strategy of contradiction.

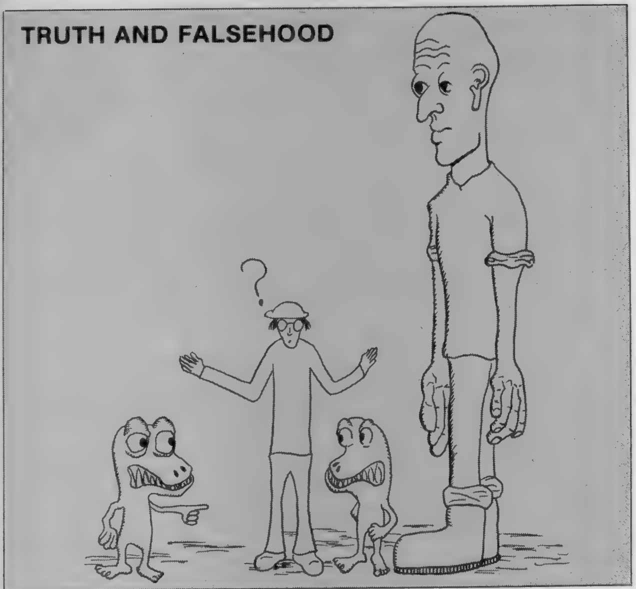
To begin with, there are two possible outcomes:

1. A finite number of natural numbers are prime or
2. An infinite number of natural numbers are prime.

No one has been able to show directly that the second alternative is true. However, by reasoning indirectly, Euclid showed that if you assume the first alternative is true, you can make a sequence of logical inferences that lead to a contradiction. Thus, the first alternative is untenable and the second must be true.

This indirect method called *reductio ad absurdum* is used throughout mathematics and represents one form of the strategy of contradiction. Interested readers may want to study Euclid's clever sequence of logical inferences where he shows that the assumption of only finitely many primes leads to a contradiction (Eves, 1976).

The following problem will be used to illustrate the method of proof by contradiction. The reader is encouraged to try the problem before reading the solution.



In a faraway land there dwelt two races. The Ananias were inveterate liars, while the Diogenes were unfailingly veracious. Once upon a time, a stranger visited the land, and on meeting a party of three inhabitants inquired to what race they belonged. The first murmured something that the stranger did not catch. The second remarked, "He said he was an Anania." The third said to the second, "You're a liar!" Now the question is, of what race was this third man?

As in Euclid's problem, there are only two possibilities. The third person is either an Anania (liar) or a Diogene (truar). Let us assume one of the alternatives, he/she is an Anania, and see if we can reach a contradiction. Since the third person said "You're a liar" to the second person and we are assuming the third person is an Anania (liar), it must follow that the second person is really telling the truth (a Diogene). But if the second person is a truar then the statement made about the first person, "He said he was an Anania," is true. But if we examine this statement closely we find that neither a liar or a truar could make it. A liar could not admit to being a liar and a truar would have to say he/she was a Diogene. Thus, we have arrived, by a sequence of logical inferences, to a contradiction based on the original assumption that the third person was a Diogene. Since there are only two alternatives, the third person must be a Diogene and the problem is solved. The reader may want to check that the problem is well-posed and that indeed, if the third person is a truar, the statements made by the first and second persons are non-contradictory.

Now we turn to an application of the method of contradiction where the choice of alternative is more than just two. Whenever the set of alternatives is small enough, it is still feasible to systematically examine each of them and derive a contradiction to all but one. As an example, consider the following problem:

WHODUNIT?

Four men, one of whom is known to have committed a certain crime, said the following when questioned by an inspector from Scotland Yard.

Growley: "Snavely did it."
Snavely: "Gaston did it."
Gus: "I didn't do it."
Gaston: "Snavely lied when he said I did it."

If only one of the four statements is true, whodunit?

Brain teasers like this one often lead the novice, who is likely to attack the problem directly, into an endless loop. The experienced problem-solver recognizes that by first assuming a particular suspect committed the crime, it is an easy matter to check whether this assumption is consistent with the given information or leads to a contradiction. For example, if we assume that Growley did it, we can determine the truth or falsity of the statements given in the problem as follows:

"Snavely did it" is false.
"Gaston did it" is false.
"I didn't do it" is true.
"Snavely lied when he said I did it" is true.

whodunit?

Since two of the suspects are telling the truth and we are given that only one of the first four statements is true, the assumption that Growley did it has lead, through a sequence of logical inferences, to a contradiction. Thus, Growley is innocent. Each suspect, in turn, can be checked out in a similar way. Whenever a problem can be reduced to a bookkeeping chore like this one, it is natural to call in the computer. Even though the work involved in this problem is small and can be easily done by hand, the ideas learned in programming the computer to do the job will be useful when we are faced with a more difficult and time-consuming problem.

We begin the WHODUNIT problem by assigning variables to statements as follows:

P(1) ⟷ "Growley did it."
P(2) ⟷ "Snavely did it."
P(3) ⟷ "Gus did it."
P(4) ⟷ "Gaston did it."

A statement is designated as true by setting the corresponding variable equal to one, P(I) = 1. If we assume it is false, then P(I) = 0. For example, if Growley is the culprit, then the statement "Growley did it" is true and P(1) = 1, otherwise P(1) = 0. Using this new representation, it is a simple matter to express all the statements (facts of the case) in one equation:

P(2) + P(4) + NOT P(3) + NOT P(4) = 1.

This expression embodies the condition that only one of the statements; "Snavely did it" P(2), "Gaston did it" P(4), "Gus didn't do it" NOT P(3), "Gaston didn't do it" NOT P(4) is true and hence the expression adds to 1. (Recall that in the BASIC language, if A = 1 then NOT A = 0 and vice versa if A = 0 then NOT A = 1.) It is a simple matter for the

computer to systematically assume each suspect, in turn, is guilty (for I = 1 to 4, P(I) = 1, and P(J) = 0, J ≠ 1) and check this assumption for any contradiction with the facts of the case, (P(2) + P(4) + NOT P(3) + NOT (P(4) = 1).

This is the technique employed in program CRIME in lines 80 to 170 to crack the case. The program also checks to see if the problem as posed has a unique solution, no solution, or perhaps many solutions. By changing line 120 (the facts of the case) a new problem can be examined with the same program. The reader is invited to make up new circumstances and test them for solutions. For example, suppose that the statements given by the suspects are: "Growley did it" P(1), "Snavely did it" P(2), "Gaston didn't do it" NOT P(4), "Snavely didn't do it" NOT P(2) and we know that only one of the suspects is lying. Then the facts of the case would be expressed as

$$P(1) + P(2) + \text{NOT } P(4) + \text{NOT } P(2) = 3.$$

Under this assumption, whodunit?

CRIME PROGRAM

```
10  PRINT "THIS PROGRAM SOLVES THE WHODUNIT PROBLEM AS ORGINALLY WRITTEN."
15  PRINT
16  PRINT "FOUR SUSPECTS - GROWLEY, SNAVELY, GUS AND GASTON - ARE QUESTIONED"
17  PRINT "ABOUT A CERTAIN CRIME. THE FOLLOWING STATEMENTS ARE GIVEN AND ONLY"
18  PRINT "ONE IS TRUE."
19  PRINT
20  PRINT "       SNAVELY DID IT."
22  PRINT "       GASTON DID IT."
24  PRINT "       GUS DIDN'T DO IT."
26  PRINT "       GASTON DIDN'T DO IT."
28  PRINT
30  PRINT "THIS IS EXPRESSED IN LINE 120 AS P(2)+P(4)+NOTP(3)+NOTP(4)=1."
32  PRINT
34  PRINT "(YOU CAN MAKE UP YOUR OWN MYSTERY BY CHANGING THE STATEMENTS"
36  PRINT "AND THE CORRESPONDING EXPRESSION IN LINE 120.)"
40  PRINT LIN(2)
50  J=1
60  DIM P[4],A$[72]
70  A$="GROWLEY DID IT.SNAVELY DID IT.GUS DID IT.    GASTON DID IT. "
80  FOR I=1 TO 4
90  MAT P=ZER
100  P[I]=1
110  REM ********************************************
120  IF P[2]+P[4]+ NOT P[3]+ NOT P[4]=1 THEN 150
130  REM ********************************************
140  GOTO 170
150  J=J+1
160  K=I
170  NEXT I
180  GOTO J OF 190,210,230
190  PRINT "SORRY THERE IS NO SOLUTION TO THIS CASE. MORE INFO IS NEEDED."
200  GOTO 250
210  PRINT "THE INESCAPABLE CONCLUSION IS THAT "A$[15*K-14,15*K]
220  GOTO 250
230  PRINT "THERE IS NO UNIQUE SOLUTION TO THIS CASE. MORE THAN ONE SUSPECT"
240  PRINT "IS IMPLICATED BY THE INFORMATION GIVEN."
250  END
```

SAMPLE RUN

```
RUN
CRIME

THIS PROGRAM SOLVES THE WHODUNIT PROBLEM AS ORGINALLY WRITTEN.

FOUR SUSPECTS - GROWLEY, SNAVELY, GUS AND GASTON - ARE QUESTIONED
ABOUT A CERTAIN CRIME. THE FOLLOWING STATEMENTS ARE GIVEN AND ONLY
ONE IS TRUE.

       SNAVELY DID IT.
       GASTON DID IT.
       GUS DIDN'T DO IT.
       GASTON DIDN'T DO IT.

THIS IS EXPRESSED IN LINE 120 AS P(2)+P(4)+NOTP(3)+NOTP(4)=1.

(YOU CAN MAKE UP YOUR OWN MYSTERY BY CHANGING THE STATEMENTS
AND THE CORRESPONDING EXPRESSION IN LINE 120.)



THE INESCAPABLE CONCLUSION IS THAT GUS DID IT.

DONE
```

cryptarithmetic

In contrast to the previous examples, where the number of alternatives were relatively small, cryptarithmetic problems usually leave a large number of possible assignments of digits to letters to be examined. For example, consider the following problem.

WIRE MONEY

A college student sent the above message to his father. If each letter represents a unique number, how much should his dad send?

In the jargon of computer science, the *search space* — the set of possible alternatives for considerations as solutions — is 8! or 40,320. It would be impractical in this problem to blindly try each possible assignment one at a time. It would be much more efficient to divide up the search space into large classes, according to a common property shared by members of each class, and then attempt to eliminate entire classes by the method of contradiction. Wickelgren (1974) has labeled this technique *classificatory contradiction.*

We can illustrate this approach with the problem above by the following argument.

Consider all possible solutions where E = 0. Since E + E = Y, Y also equals 0, contradicting the fact that Y and E must be different digits. Thus, the entire class of solutions where E = 0 is ruled out. For another, less trivial example, consider E = 3. Now Y equals 6 and there is no carry to the next column. Thus in the record column we have R + R = E or perhaps R + R = E + 10 if a carry is involved. But in either case, E would be an even number, since 2R is always even, and this contradicts the assumption that E = 3. Thus the entire class of solutions with E = 3 is ruled out.

By following this same type of classificatory contradiction strategy for each of the digits in order of E, R, I, O, and N, program CRYPT search out all five solutions to the WIRE + MORE = MONEY problem. Can you write your own program to solve the cryptarithmetic problem.

$$\begin{array}{r} \text{DONALD} \\ +\ \text{GERALD} \\ \hline \text{ROBERT}\quad ? \end{array}$$

This problem has been extensively studied by Newell and Simon (1972) using a program called General Problem Solver (GPS) which was written to demonstrate that general problem-solving strategies exist and may be discussed at the very concrete level of computer programming.

CRYPT PROGRAM

```
10  PRINT "THIS PROGRAM SEARCHES OUT ALL SOLUTIONS"
20  PRINT "TO THE CRYPTARITHMETIC PROBLEM"
30  PRINT
40  PRINT "        W     I     R     E"
50  PRINT "    +   M     O     R     E"
60  PRINT "-----------------------------"
70  PRINT " M      O     N     E     Y"
80  PRINT LIN(2)
90  M=1
100  FOR E=2 TO 9
110  Y=E+E
120  IF Y >= 10 THEN 150
130  C1=0
140  GOTO 170
150  C1=1
160  Y=Y-10
170  FOR R=0 TO 9
180  IF R=M OR R=E OR R=Y THEN 470
190  IF R+R+C1=E THEN 220
200  IF R+R+C1=E+10 THEN 240
210  GOTO 470
220  C2=0
230  GOTO 250
240  C2=1
250  FOR I=0 TO 9
260  IF I=M OR I=E OR I=Y OR I=R THEN 460
270  FOR O=0 TO 9
280  IF O=M OR O=E OR O=Y OR O=R OR O=I THEN 450
290  N=I+O+C2
300  IF N >= 10 THEN 330
310  C3=0
320  GOTO 350
330  C3=1
340  N=N-10
350  IF N=M OR N=E OR N=Y OR N=R OR N=I OR N=O THEN 450
360  FOR W=0 TO 9
370  IF W=M OR W=E OR W=Y OR W=R OR W=I OR W=O OR W=N THEN 440
380  IF O÷10#W+M+C3 THEN 440
390  PRINT "        ";W;I;R;E
400  PRINT "        ";M;O;R;E
410  PRINT "-----------------------------"
420  PRINT M;O;N;E;Y
430  PRINT LIN(2)
440  NEXT W
450  NEXT O
460  NEXT I
470  NEXT R
480  NEXT E
490  END
```

SAMPLE RUN

```
RUN
CRYPT

THIS PROGRAM SEARCHES OUT ALL SOLUTIONS
TO THE CRYPTARITHMETIC PROBLEM

         W     I     R     E
     +   M     O     R     E
------------------------------
 M       O     N     E     Y


         9     7     6     2
         1     0     6     2
------------------------------
 1       0     8     2     4


         9     2     7     4
         1     0     7     4
------------------------------
 1       0     3     4     8


         9     5     7     4
         1     0     7     4
------------------------------
 1       0     6     4     8


         9     2     8     7
         1     0     8     7
------------------------------
 1       0     3     7     4


         9     5     8     7
         1     0     8     7
------------------------------
 1       0     6     7     4


DONE
```

conclusion

When it comes to problem solving, one of the most significant advantages that people have over computers is the ease in which humans can make inferences about a problem and quickly reduce the search space. Programming a computer to make similar inferences is at best extremely difficult and at worst, impossible. People can be very unpredictable in ways to survey a problem looking for the easy inferences to attack first, while the computer is easiest to program to attack each problem in a very predetermined and algorithmic way. Polga (1954) contrasts the algorithmic method with the *heuristic* approach in which the nature of the solution is guessed and then proved to be correct. These two approaches have been combined into "heuristic programming" which is a new way of thinking about what a computer program should do. The idea is to find a good set of rules for generating guesses and then prove they are correct. This technique evolved from the work of Newell and Simon (1972) working on the program Logic Theorist (LT) and General Problem Solver (GPS).

The reader can appreciate the difficulty of heuristic programming by imagining how one would write a program to solve general cryptarithmetic problems in which the simplest inferences in a problem are first made similar to the way a human would approach it. In contrast a general algorithmic program can be written to solve any cryptarithmetic problem in a way similar to program CRYPT. Can you do it?

References

Carroll, Lewis. *The Humorous Verse of Lewis Carroll,* Dover Publications, New York, NY, 1960.

Eves, Howard. *An Introduction to the History of Mathematics,* Holt, Rinehart, and Winston, New York, NY, 1976, p. 120.

Newell, A., and Simon, H. *Human Problem Solving,* Prentice Hall, Englewood Cliffs, NJ, 1972.

Wickelgren, Wayne A. *How to Solve Problems,* W.H. Freeman and Company, San Francisco, CA, 1974.

Acknowledgement

We are extremely grateful to Rodney Schroeter for the drawings which he provided throughout the series. ■

IFIP Congress 77

The Exhibits Were as Interesting as the Sessions

Ruth Glick

Computer conventions are all the same. The papers and panel discussions are the ostensible reasons for attending. But the hardware and software displays mounted by various companies and organizations are a big part of the attraction.

The International Federation of Information Processing Societies' Congress 77—held during August 1977 in Toronto—was no exception. Convention-goers jammed the exhibit halls to find out what more than 100 large and small computer companies, book and magazine publishers, and various Canadian government agencies had to offer.

For their part, the exhibitors put on quite a show. Graphics displays and data readouts flashed on terminal screens, tape-drives spun, printers spewed out copy, salesmen explained the virtues of their particular equipment and services, and convention-goers interacted with demonstration programs.

There was so much going on that catching the eye of spectators was important. Some organizations used clever gimmicks to draw the crowds. Others, particularly those with unusual computer graphics displays were interesting enough to capture attention on their own merits.

Public Works Canada, for example, had a display of computer-assisted drafting. According to Lew Woolsey, who was demonstrating the equipment, the system is much faster and more cost-effective than conventional drafting. Yet even for someone with no computer experience, it takes less than a week to learn. Using the display terminal, architects can try various designs. draw accurate lines "automatically," and zoom in on portions of a building as they work. They can have the computer display buildings with designs similar to the one they are planning, try out various features like windows and doors, convert from conventional measurements to metric, and change the scale of drawings at will.

"You can put the catalog from a window company in the computer," Woolsey said, "and try out 27 different types of windows to see how they'd look .... The system takes the dog work out of drafting, and it reduces the chances for costly errors." Besides the graphics capability, the system can also run stress analyses, calculate heating and cooling losses, and specify requirements for lighting, mechanical ductwork, piping, and the like.

IFIP Sessions Cover Job Satisfaction, Future Communications, More.

While the sessions at IFIP were, for the most part, not of direct interest to *Creative* readers, at least two presented some revelant material

One panel looked across a wide variety of industries and concluded that the use of computers to perform data-processing functions deprives workers of job satisfaction. Examples were cited in banks, machine shops, and steel mills. Prior to using teller terminals, tellers used to suggest changes that would improve customer services; their attitudes paralleled those of the bank managers. After computerization, their suggestions concerned coding schemes, new forms or new displays. Within a year tellers behaved more like keypunch operators than bank managers. Other panelists described EDP systems that had been installed in other industries which took little or no account of the psychological attitudes of the affected workers.

At the session "Data Networks — Past, Present and Future" Paul Baron of Cabledata Associates speculated that future networks would increasingly be based on individuals using micro-based packet switching rather than more growth by governments and large corporations. He felt that a people-to-people-type structure gives society a more stable organization and one that is more adaptive to change than a monolithic hierarchy as exists in the government today.

Baron sees five components being necessary for such people-to-people packet-switched systems: a user terminal, a local distribution system, a packet-switching node, long-distance transmission lines or satellites, and a host computer. As a result of rapidly declining prices, microprocessor-based nodes are now affordable by hobbyists. Baron's suggestions for transmission mostly involved "bootlegging onto phone lines or satellites without detection." [Pub. Note: It's not clear that such an illegal approach could really become widespread.] Baron felt that the main stumbling block to increased interpersonal (data) communication, particularly over international boundaries, is government control. Perhaps the channels should be open to individuals he suggested. [Pub. Note: Once in place, government regulations tend to be extremely difficult to change but there may well be other alternatives such as the AMSAT satellite or the much-rumored Bell Data Network.]

Personal-computing topics were touched on in other sessions but in nothing close to the depth of the Faire in SF, NCC in Dallas, or PC '77 in Atlantic City. —DHA

Quite different was a display set up by the Canadian National Institute for the Blind, where several sightless programmers were demonstrating braille terminals manufactured by Triformation Systems of Stuart, Florida (the only makers of this equipment in North America). David Brown, Manager of Computer Services for the Institute, explained that the braille printout from the machines is used by these programmers in much the same way that sighted programmers use the screen of a display terminal. "In the old days, they had to have computer output read to them. Now they can work independently," he said.

At the Institute's computer department, the staff employs the equipment to do payroll and keep records on clients, eye diseases, and causes of blindness.

"People are reluctant to hire the blind, because they think that extra problems will be created. We'd like to convince them that blind workers can be just as effective as sighted ones," he said. Presently, he noted, there are about 75 blind programmers employed across Canada. All have been through a special training course at the University of Manitoba where they learn to make flowcharts in braille on individual cards as well as use the Triformation terminals.

However, the equipment is not designed to be used exclusively by programmers. Many blind workers, including credit and reservations clerks, various people in the telecommunications field, stockbrokers, engineers and radio broadcasters, have been using the terminals. And Brown hopes more employers will realize the potential for the blind.

There were also other organizations on hand with equipment for special users. Systemhouse Limited was showing a computer-assisted cartography package, which can be used in mapmaking much as the computer drafting system displayed by Public Works Canada is used in building design. Ron Speer, who was demonstrating the equipment, put up a map of Canada on the screen and then had the computer enlarge a portion of it several times. "You can get down to one-meter ground resolution," he explained. The equipment, which has been purchased from Systemhouse by the Australian Army, is especially useful as a map editing tool, because you zoom in on boundary areas (where two maps have been put together) and touch up inconsistencies. Speer demonstrated by drawing a river using a joy-stick on the console.

A number of companies were also selling computer equipment for hospital use. One, Siemens, was showing a medical reporting system for diagnostic X-ray departments. Their terminal was especially interesting because it was designed to be activated by touching appropriate sections of an oversized viewing screen rather than by typing on the keyboard. The large screen can display any one of 200 frames depending on the injury that the physician or X-ray technician needs to describe. Information entered on the large screen is displayed as a medical report on a smaller screen and can be printed out as hard copy to be added to the patient's file.

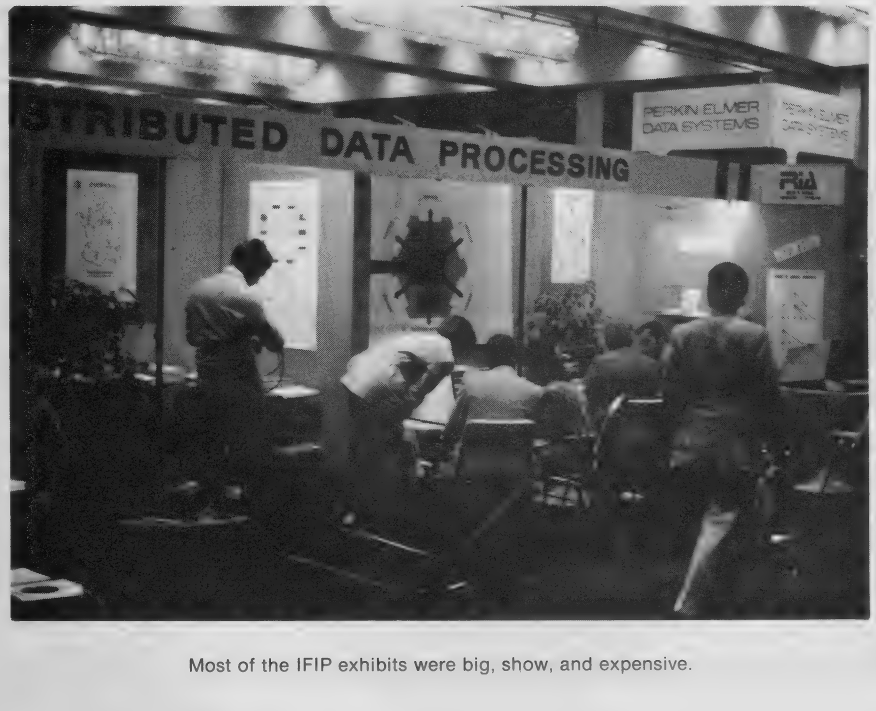
Other exhibits were attention getting because of the gimmicks they employed.

Probably the most popular display at the convention was the one belonging to MSA (Management Science America, Inc.), an employee-owned company based in Atlanta, Georgia, "in the business of designing, developing, servicing, and marketing a complete line of financial accounting software systems."

To attract the attention of prospective customers, MSA's Jerry Bertsche had set up a computer portrait studio where long lines of convention-goers waited during exhibit hours each day to

Servant or Master?

The computerized message system at IFIP operated by "The Computer Group" of the TransCanada Telephone System posted names of people for whom telephone messages had been received on strategically-located CRT screens. Of course, as more and more messages accumulated, the longer one had to wait while the system scrolled thru the whole list. It printed at about 300 baud and then left a full screen of only 8 (large) lines up for 10 seconds. Hence, to get through the whole list by the 3rd day of the conference took about 4 minutes. But why, you may ask, was the list getting longer? Because most of the telephones to retrieve your message didn't work! (It took me over 10 minutes to retrieve a message which, had it been posted on a bulletin board in alphabetical order, would have taken less than 10 seconds). Ah, technology — who is the servant and who is the master? —DHA



Most of the IFIP exhibits were big, show, and expensive.

IFIP Congress 77

Do computer portraits sell software? Apparently MSA thinks so, and they certainly always had a crowd at their booth.

have their "pictures taken." Designed by Computer Concepts Corporation, the system consists of a television camera, monitor, computer and printer. According to the company's literature, "the camera sends the computer a picture in the form of the black and white images that it 'sees.' The computer converts these images to the appropriate typewriter characters and prints them." And the convention-goer leaves with an interesting souvenir—his or her very own computer-generated portrait.

Another well-attended "entertainment" at the convention was presented by Sperry Univac, which wanted to impress attendees with the company's high-volume timesharing capabilities. To attract attention they put on a twenty-minute show every hour and a half featuring (what else!) a pretty girl and an irreverent televised cartoon character named Homer Highwater. As it turned out, Homer was more of a draw than the girl. Instead of just delivering a canned presentation, he actually interacted with the audience—asking people's names, making faces, responding to questions, flailing his arms about and adlibbing like mad. Some of his repartee was designed to get a laugh at the expense of the spectators. For example, during one presentation he accused a man standing near his pretty assistant of mentally undressing her instead of paying attention to the Sperry Univac pitch. Then he turned to *Creative Computing's* reporter and announced that she must be "sick," because she was taking notes on his jokes.

Sperry Univac wouldn't say exactly how Homer did his real-time tricks. But they did admit that he wasn't generated by a computer. Apparently a live human being talks for Homer and manipulates his cartoon image in some way.

At another exhibit, SDL (Systems Dimensions Limited) was offering a Toronto restaurant selection service. Convention-goers could request a list of establishments in a particular price range or those offering a certain type of cuisine and receive a computer printout of appropriate restaurants. For a less practical demonstration of the SDL equipment, they would have their biorhythms calculated by the company's WYLBUR EXEC files.

And finally there were the inevitable computer games. At Congress 77 they were provided at the booth set up by *Interface Age* magazine. Spectators could select an assortment of games—from Life and Chess to Moon Landing, Star Trek and Target—and play them on a TV monitor attached to a microcomputer. The games were stored on programmed diskettes.

In all, there was a lot going on in the exhibit halls—as much as in panel discussions and other official presentations. ■

The Future of Computing

What Do YOU Think?

We would like you to help us explore the future of computing as part of a study for *Creative Computing* magazine. The questionnaire below contains three sections. The first asks for some basic background information about you, your contact with computers, and your general views about the future. The second section contains statements about developments related to computing which may or may not occur in the next twenty-five years.

We would like you to rate each item in terms of how likely you feel its occurrence will be, how important it will be if it occurs, and how desirable it will be if it occurs. These hypothetical results were generated in part by the *Creative Computing* readership and in part by reference to previous studies of the future of computing. Some statements may seem overly vague, others too specific. Please try to evaluate each in terms of the general trend it represents. The final section of the questionnaire asks for your ideas, comments and suggestions on the future of computing. Please feel encouraged to elaborate on them.

When you've completed the questionnaire, please mail the results to:

Craig Johnson
The DaVinci Group
235 Oak Drive
Willowdale Lake
North Canton, Ohio 44720

QUESTIONNAIRE

Section I

(1) ____Age

(2) Sex
____1. Female
____2. Male

(3) Education
____1. Less than high school graduate
____2. High-school graduate
____3. Some college
____4. Bachelor's degree
____5. Master's degree
____6. Doctoral degree

(4) How much background or training in computer science do you have?
____1. No background
____2. Slight background
____3. Significant background (some formal training)
____4. Very substantial background (professional employment or professional degree in computer science)

(5) About how often have you used computers directly in the last year?
____1. Rarely (a few times a year or less)
____2. Occasionally (once a month)
____3. Frequently (once a week)
____4. Daily

(6) On a scale from 1 to 5, how would you describe your political beliefs?
____1. Very conservative
____2. Conservative
____3. Middle of the road
____4. Liberal
____5. Very liberal

(7) How do you feel about your *personal* future?
____1. Very pessimistic
____2. Somewhat pessimistic
____3. Neutral, can't say
____4. Optimistic
____5. Very optimistic

(8) How do you feel about the future of the *United States?*
____1. Very pessimistic
____2. Pessimistic
____3. Neutral, can't say
____4. Optimistic
____5. Very optimistic

(9) How do you feel about the future of the *world?*
____1. Very pessimistic
____2. Pessimistic
____3. Neutral, can't say
____4. Optimistic
____5. Very optimistic

(10) Do you think forecasting studies are useful?
____1. Useless
____2. Of some use
____3. Very useful

Section II

This section lists a number of events which may or may not occur in the next twenty-five years. Please rate each event in three ways, on a scale of 1 to 5:

How *likely* is it?
1. Very unlikely, almost impossible
2.
3.
4.
5. Very likely, almost certain

How *important* will it be if it occurs?
1. Very unimportant, trivial
2.
3.
4.
5. Very important, crucial

How *desirable* will it be if it occurs?
1. Very undesirable
2.
3.
4.
5. Very desirable

(1) Audio (spoken) communication will be a common input/output mode.
____Likelihood ____Importance ____Desirability

(2) Most governmental decisions will be made by debate and opinion analysis over a computer network.
____Likelihood ____Importance ____Desirability

(3) Holographic (three dimensional) audio/visual output will be in common use.
____Likelihood ____Importance ____Desirability

(4) Problems of poverty, population growth and environmental decay will be largely solved for the United States.
____Likelihood ____Importance ____Desirability

(5) Costs of computing (costs per instruction executed) will decrease by a factor of 100 from current (1977) levels.
____Likelihood ____Importance ____Desirability

(6) Fear of invasion of privacy and general technical and economic problems will prevent the development of large data "nets" (networks).
____Likelihood ____Importance ____Desirability

(7) Direct two-way brain/computer (biocybernetic) links will be common.
____Likelihood ____Importance ____Desirability

(8) Intelligent, self-aware computers (capable of passing the Turing test) will exist.
____Likelihood ____Importance ____Desirability

(9) Complex interactions with computers will be carried out by a mix of audio communications and movement through a sensor field (programming by song and dance).
____Likelihood ____Importance ____Desirability

(10) Diffusion of information and authority throughout large computer-based bureaucracies will produce widespread alienation.
____Likelihood ____Importance ____Desirability

(11) Basic concepts of computer science will be taught in elementary school.
____Likelihood ____Importance ____Desirability

(12) Almost all financial transactions will be carried out by computer with no physical exchange of money.
____Likelihood ____Importance ____Desirability

(13) Problems of poverty, population growth and environmental decay will be largely solved for the world.
____Likelihood ____Importance ____Desirability

(14) Integrated teams of artists and scientists using computer communications and data-handling will be used to attack major social and environmental problems.
____Likelihood ____Importance ____Desirability

(15) The majority of American homes will have a computer console.
____Likelihood ____Importance ____Desirability

(16) Costs of rapid, randomly-accessible memory (cost per bit accessed) will decrease by a factor of 10,000 from current (1977) costs.
____Likelihood ____Importance ____Desirability

(17) Rapid increase in automation will generate large numbers of unemployed workers.
____Likelihood ____Importance ____Desirability

(18) Most art and entertainment will be generated and viewed via computer.
____Likelihood ____Importance ____Desirability

(19) Costs of rapid, randomly-accessible memory (cost per bit accessed) will decrease by a factor of 100 from current (1977) costs.
____Likelihood ____Importance ____Desirability

(20) A very large "computer hobbyist" industry will develop.
____Likelihood ____Importance ____Desirability

(21) The United States will no longer dominate global economic and political affairs.
____Likelihood ____Importance ____Desirability

(22) Pocket computers will have independent capacity equal to current "third generation" computers (IBM 360, Burroughs 6500, UNIVAC 1108, etc.)
____Likelihood ____Importance ____Desirability

(23) Computer/communications utilities will become the largest industry (in dollar volume of transactions) in the U.S.
____Likelihood ____Importance ____Desirability

(24) Most major household appliances will contain microcomputers to operate them in home use.
____Likelihood ____Importance ____Desirability

(25) Pocket computers will be capable of automatically linking to computing "nets" (networks) if they are within a mile of a two-way telecommunications channel.
____Likelihood ____Importance ____Desirability

(26) The majority of U.S. computers will be linked into a general computing/memory network.
____Likelihood ____Importance ____Desirability

(27) Most governmental and business decisions will be made directly by computer with little human intervention.
____Likelihood ____Importance ____Desirability

(28) Computer-based data gathering and analysis systems will provide the basis for rapid advances in the social and environmental sciences.
____Likelihood ____Importance ____Desirability

(29) Costs of computing (cost per instruction executed) will decrease by a factor of 10,000 from current (1977) costs.
____Likelihood ____Importance ____Desirability

(30) Pocket-sized memory units will have capacity equal to contemporary disk memories (will have capacities of approximately 1 billion bits per cubic inch).
____Likelihood ____Importance ____Desirability

(31) Libraries with "hard copy" (books, etc.) will be largely replaced by computer-based data files.
____Likelihood ____Importance ____Desirability

(32) Automatic translators for natural (human) languages will be in common use.
____Likelihood ____Importance ____Desirability

(33) Computers will exist which can comprehend standard intelligence (IQ) tests and score over 200 on them.
____Likelihood ____Importance ____Desirability

(34) Computers which "learn" from experience and are "educated" rather than programmed will be in common use.
____Likelihood ____Importance ____Desirability

(35) Breakdowns or errors in computer-controlled systems will cause several disasters of serious proportions (resulting in hundreds of deaths or injuries).
____Likelihood ____Importance ____Desirability

(36) Computer-based job-search procedures will reduce unemployment and under-employment.
____Likelihood ____Importance ____Desirability

(37) Computer hardware will be largely based on biological and biochemical circuitry.
____Likelihood ____Importance ____Desirability

(38) The health of the U.S. population will improve because of computer-based diagnostic and health-monitoring techniques.
____Likelihood ____Importance ____Desirability

(39) The overall quality of life for the average American will be greatly improved.
____Likelihood ____Importance ____Desirability

(40) The overall quality of life for the average human will be greatly improved.
____Likelihood ____Importance ____Desirability

Section III

Please give us any comments, suggestions or ideas you may have related to the future of computing, the future of the U.S. or the future of the world. ■

Mondrian has been widely acclaimed as the greatest Dutch painter of our time and as one of the most influential masters of painting. However, a computer-generated random pattern was preferred over the pattern of one of Mondrian's paintings. Why?

HUMAN OR

AESTHETIC FOR PSEUDO COMPUTER PATTERNS

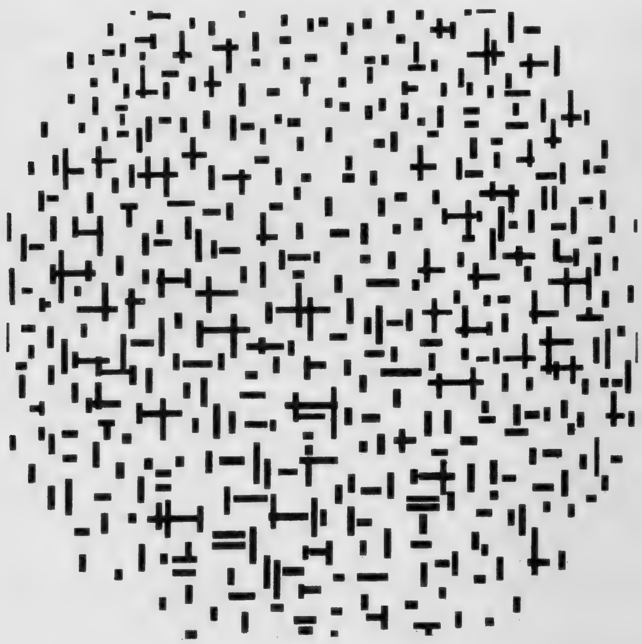


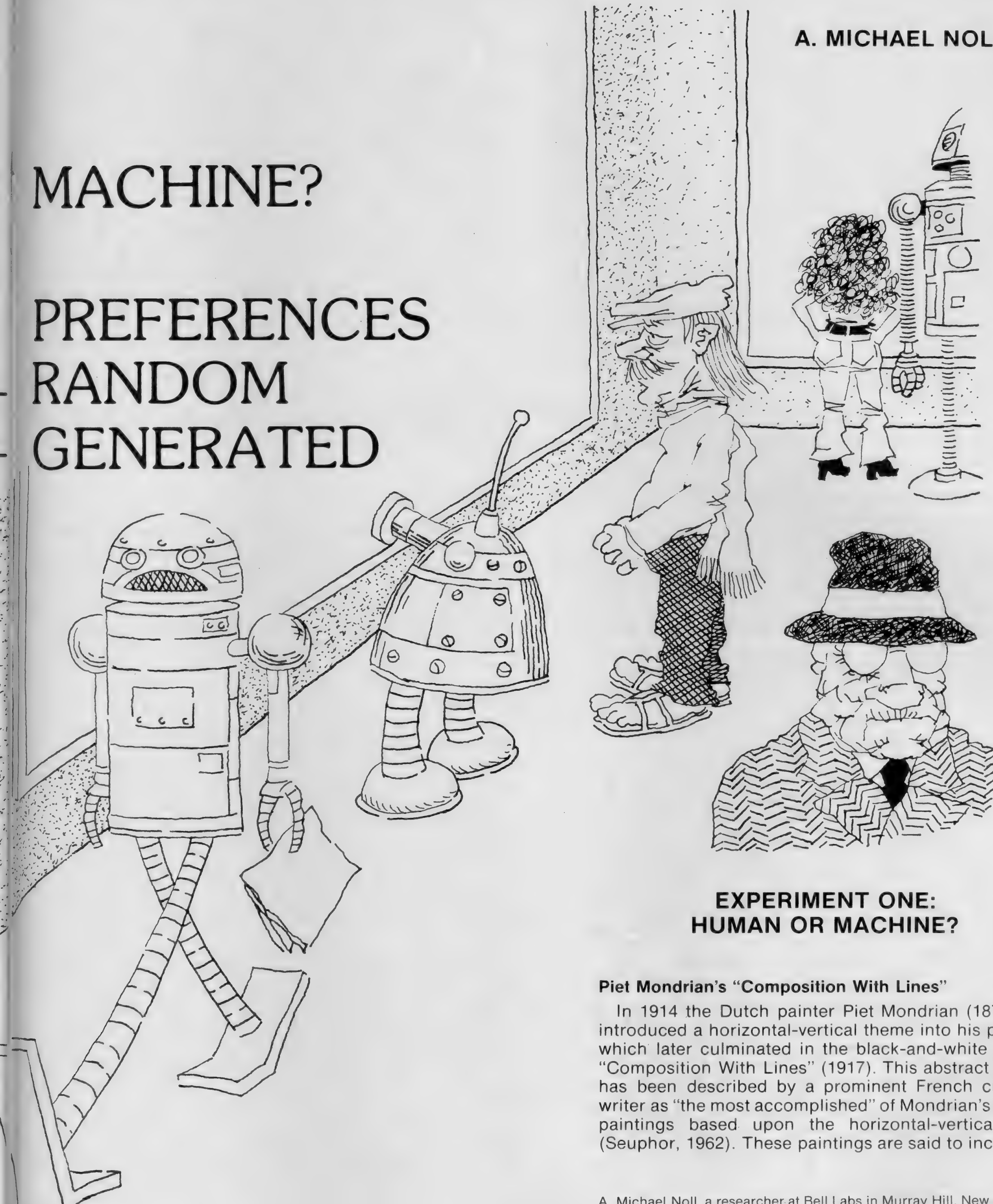
Fig. 1

Fig. 1 "Composition With Lines" (1917) by Piet Mondrian. (Reproduced with permission of Rijkmuseum Kroller-Muller, Otterlo, The Netherlands, c Rijkmuseum Kroller-Muller.)

MACHINE?

PREFERENCES
RANDOM
GENERATED

A. MICHAEL NOLL



EXPERIMENT ONE: HUMAN OR MACHINE?

Piet Mondrian's "Composition With Lines"

In 1914 the Dutch painter Piet Mondrian (1872-1944) introduced a horizontal-vertical theme into his paintings which later culminated in the black-and-white painting "Composition With Lines" (1917). This abstract painting has been described by a prominent French critic and writer as "the most accomplished" of Mondrian's series of paintings based upon the horizontal-vertical theme (Seuphor, 1962). These paintings are said to incorporate

A. Michael Noll, a researcher at Bell Labs in Murray Hill, New Jersey, is a pioneer in the use of computers in the visual arts and in experimental aesthetics. In this article, he describes his classic experiment in which a computer-generated picture was preferred to a Mondrian painting. The article is an abridgement of material previously published in *The Psychological Record* [Vol. 16, pp. 1-10 and Vol. 22, pp. 449-462].

masculinity and femininity by symbolizing the masculine as vertical (the upright trees of a forest) and the feminine as horizontal (the sea) with each complementing the other (Seuphor, 1957). Mondrian sought to indicate the plastic function of the sea, sky, and stars through a multiplicity of crossing verticals and horizontals (Mondrian, 1945). "Composition With Lines," shown in Fig. 1, consists of a scattering of vertical and horizontal bars which, at first glance, seem to be randomly scattered throughout the painting. With further study, however, one realizes that Modrian used considerable planning in placing each bar in proper relationship to all the others. Conceivably, Mondrian followed some scheme, or program, in producing the painting although the exact algorithm is unknown.

If Mondrian's "Composition With Lines" is studied carefully, some interesting observations about its overall composition can be made. The more evident of these are: (a) The outline of the painting is a circle that has been cropped at the sides, top, and bottom; (b) The vertical and horizontal bars falling within a region at the top of the painting have been shortened in length; and (c) The length and width of the bars otherwise seem to be randomly distributed.

"Computer Composition With Lines"

Many pictures can be thought of as consisting of series of connected and disconnected line segme ts. Since two points determine a line, such pictures can be described numerically by the cartesian coordinates of the end points of the lines. Thus, a picture can be transformed into numerical data which are then inversely transformable back into the original picture.

Digital computers perform arithmetic operations with numerical data under the control of a set of instructions called a program. If this numerical data were the coordinates of end points of lines, then the computer could be programmed to numerically specify a picture. This numerical data could then be used to position and move the beam of a cathode ray tube to trace out the desired picture. In this manner, an IBM 7094 digital

© AMN 1965

Fig. 2 "Computer Composition With Lines" (1964) by A. Michael Noll (c A. Michael Noll 1965.)

computer was programmed to generate pictures using a General Dynamics SC-4020 Microfilm Plotter. The picture drawn on the face of the cathode ray tube was photographed by a 35 mm camera which was also under the control of the microfilm plotter.

Mondrian's "Composition With Lines," a black and white painting composed of vertical and horizontal bars, was a type picture that the microfilm plotter was capable of reproducing with suitable programming of the computer. The computer picture thus generated called "Computer Composition With Lines," is shown in Fig. 2.

The vertical and horizontal bars in "Computer Composition With Lines" were produced as a series of parallel line segments that were closely enough spaced to slightly overlap each other. Although Mondrian apparently placed his bars in a very-orderly manner, the computer was programmed to place the bars randomly within a circle of radius 450 units so that all locations were equiprobable. The choice between vertical bar or horizontal bar was equally likely, and the widths of the bars were equiprobable between 7 and 10 lines; the lengths of the bars were equiprobable between 10 and 60 points.

If a bar fell inside a parabolic region at the top of the picture, the length of the bar was reduced by a factor proportional to the distance of the bar from the edge of the parabola. A trial-and-error approach was used to insure that the effect of the picture was similar to Mondrian's "Composition With Lines."

Human or Machine?

After the computer had produced its version of the Mondrian painting, two pictures similar in composition, but one painted by a human and the other generated by a machine, were available. Subjective tests were then administered in which subjects were shown reproductions of both pictures and indicated their preferences and also which picture they thought was produced by the machine.

Procedure

The photographic print of the computer-produced microfilm and the photograph of Mondrian's painting had clues to their identity since the quality of the photographs was somewhat different. Since only differences in the designs or patterns of the two pictures were desired, the two photographs were copied xerographically to be identical in quality. These copies were arranged in two pairs so that the computer picture was alternately labeled "A" or "B"; the order of presentation was counter-balanced. An example of a picture pair as given to the 100 subjects who participated in the experiment is shown in Fig. 3.

In addition to the two pictures, each subject was also given two questionnaires: an identification questionnaire and a preference questionnaire. The identification questionnaire was worded: "One of the pictures is a photograph of a painting by Piet Mondrian while the other is a photograph of a drawing made by an IBM 7094 digital computer. Which of the two do you think was done by the computer." The subject then checked appropriate boxes on the questionnaire for picture "A" or for picture "B" and also gave written reasons for his choice. The preference questionnaire asked the subject to check appropriate boxes to indicate which picture he "most strongly liked or preferred" and also to give reasons for his choice. The order of presentation of the identification and preference questionnaires was counter-balanced.

The 100 subjects who participated in the experiment were all employees of a large industrial research laboratory in New Jersey.

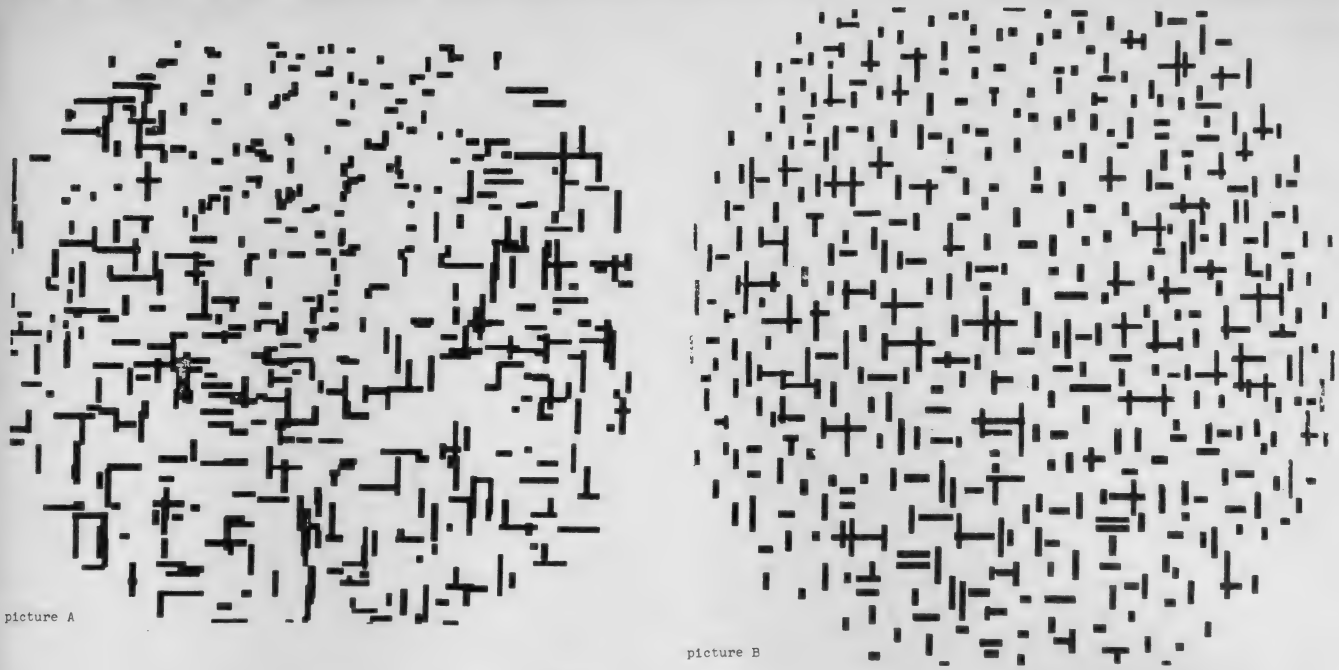


Fig. 3 Picture pair as presented on separate sheets to subjects. The original microfilm and Mondrian photographed were copied xerographically and then reproduced so that both pictures were identical in quality. (c A. Michael Noll 1965 and c Rijkmuseum Kroller-Muller.)

Results

Of the 100 subjects in the experiment, 59% preferred the computer picture while only 28% were able to correctly identify the computer picture. Both percentages were statistically different from selections based upon chance.

Perhaps the subjects' preferences would be affected by the knowledge that one of the pictures was generated by a computer. For this reason the first fifty subjects were given the preference questionnaire first while the second fifty were given the identification questionnaire first. Appropriate statistical tests (chi-square with Yates' correction) indicated that the subjects' preferences were not affected by knowledge that one of the pictures was computer generated.

The reasons given by the subjects for both their preferences and identifications were frequently quite intriguing. The computer picture was described as being "neater," more "varied," "imaginative," "soothing," and "abstract" than the Mondrian. One subject even found some golden rectangles in the random designs within the computer picture.

The knowledge that one of the pictures was produced by a computer did not bias the subjects for or against either picture, as mentioned previously. However, the subjects in this experiment had very little or no artistic training and also were quite accustomed to the impact of technology upon many different fields. These subjects therefore probably did not have any prejudices against computers as a new artistic medium. If artists and subjects from a nontechnological environment had been similarly tested, the results might have been different.

EXPERIMENT TWO: ARTIST VS NONARTIST

The results of the Mondrian experiment led to two questions. The computer-generated picture looked "more random" to subjects than the Mondrian painting; therefore, a question arose about the aesthetic preferences for a sequence of patterns varying in some attribute from complete order to complete disorder. The subjects in the Mondrian experiment had little or no artistic training; therefore, a question arose about the aesthetic preferences of artists versus nonartists.

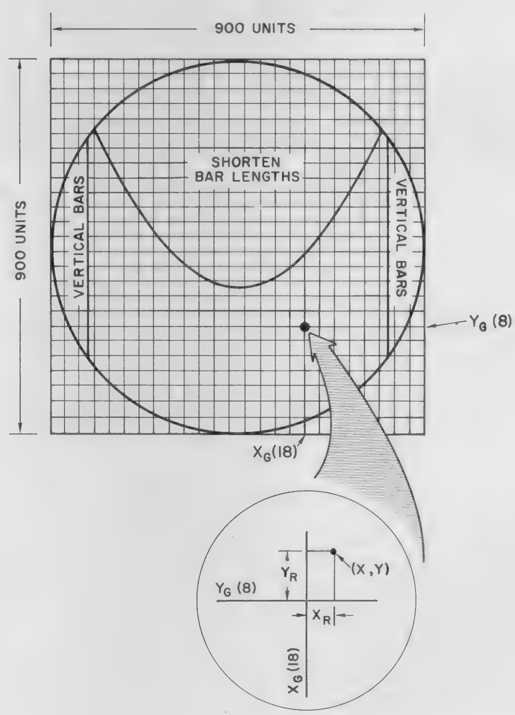


Fig. 4 Uniform grid structure and underlying computer-generated pseudorandom pictures. Random perturbations were added to the X and Y coordinates of each of the grid intersections. For example, if XG(18) and YG(8) represent the coordinates for an intersection, then two pseudorandom numbers, XR and YR, with uniform probability density and specified range, R, are added to the grid coordinates to produce the final coordinate pair where a bar is plotted.

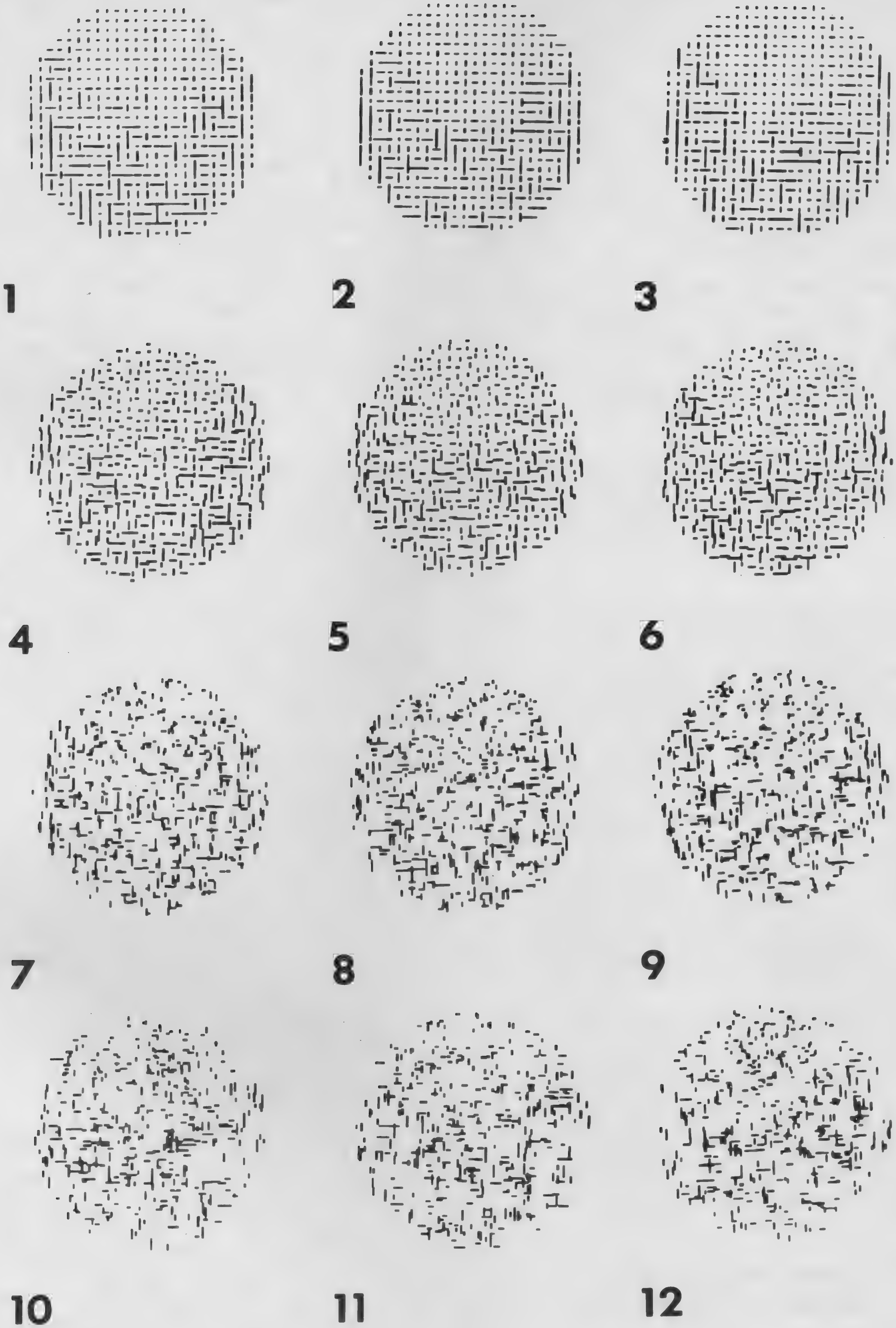


Fig. 5 The 12 computer-generated pictures used to study artistic preferences. The three pictures in any single row are statistically identical. Four ranges of bar perturbations (0, 10, 50, and 250 units) were used to produce the four sets of three statistically identical pictures.

These two questions led to a second experiment in which only computer-generated pictures were used as stimuli for a homogeneous subject population of undergraduate students differing only in their artistic training. The experiment was designed to determine: (a) the scale of aesthetic preference for different ranges of randomness in pseudorandom patterns, (b) the differences, if any, in preferences for statistically identical pseudorandom patterns, and (c) the differences, if any, between the aesthetic preferences of subjects with and without artistic training.

Stimuli

A digital computer was again programmed to plot a series of patterns consisting of vertical and horizontal black bars that varied in placement, length, and width. The bars were constrained to fall within a circular region with a predominance of vertical bars at both sides of the region. The bars falling within a parabolic region at the top of the circular region were made increasingly shorter as their distance from the boundary of the parabola increased, as shown in Fig. 4. All bars were placed at the intersections of random perturbations from a uniform grid. A series of pictures ranging from a completely uniform bar placement to a very random bar placement was produced by varying the range of the perturbations from zero to a maximum.

Four different ranges, 0, 10, 50, and 250 units, were specified for the perturbations. Three patterns were calculated by a digital computer and plotted automatically for each range; the three patterns for each range were statistically identical. A total of 12 patterns were thus produced and are shown in Fig. 5. Although the three pictures generated at each range varied in actual bar placement depending upon the pseudorandom numbers applied, they were identical in a statistical sense, since they were all produced from the same pseudorandom process.

Glossy photographic prints were made of all 12 computer-generated patterns. Multilith mats and prints were then made of all 66 possible pairs in both AB and BA orderings. The multilith prints were placed into 20 loose-leaf binders. Each binder contained all 66 possible pairs, and a different random order of the sequence of the pairs and their AB order was used for each binder.

Procedure

A combined instruction sheet and background questionnaire was distributed to the subjects. The instructions were: "This is a study to investigate the aesthetic appeal of various patterns. The booklet contains 66 pairs of pictures. Your task is to decide which picture of each pair is more appealing to you." The questionnaire requested each subject to give his name, age, sex, marital status, occupation, major field of specialization or study and highest completed education or degree. Each was also asked to "briefly describe any artistic training or art courses that you have had."

The experiment was conducted at a small university in New Jersey. The subject population consisted of 124 undergraduate students (53 males and 71 females) at the university. The university attracted a rather homogeneous student body; therefore, the subjects could be assumed to have similar socioeconomic backgrounds.

The subjects' descriptions of any artistic training or art courses they may have taken were used to classify their artistic training as (a) none or (b) some or substantial. The quantity and depth of art courses were considered along with any substantial self-study or private training. The subjects placed in the no-artistic-training category had taken no art courses; they henceforth will be called the nonartistic-trained subjects or for short, the "nonartists." Thirty-nine subjects were classified as artists based upon their artistic training; 85 were classified as nonartists.

Results

A number of different mathematical models and statistical methods were used to analyze the data to determine answers to the questions posed in the introduction to this experiment.

If the individual differences between the psychological scales of the subjects are assumed to be negligible, then an "average" psychological scale can be determined for the whole subject population. Figure 6 shows this single, unidimensional, underlying, "average" scale (determined by a Case V solution of Thurstone's law of comparative judgment). The three patterns corresponding to a bar-perturbation of 50 units were most preferred, according to this solution.

Perhaps the most surprising result of the experiment was the large variability among the subjects in their preferences for the pictures. The answer to the question of whether there is a single scale of preference for randomness appears to be that there is not one scale but many scales and possibly as many scales as there are subjects (or permutations of stimuli). Thus, a single scale was not really appropriate for the data.

The results show that most subjects judged the stimuli on the basis of randomness, and for them the statistically identical pictures were equally preferred. However, a small group of subjects did not judge according to randomness but characteristically expressed preferences between pictures of the same degree of randomness. Although the judgments of these subjects could not be related to some other physical property, their very existence suggests that statistically identical pictures could include subtle differences for some viewers.

As a group, the aesthetic preferences of subjects with

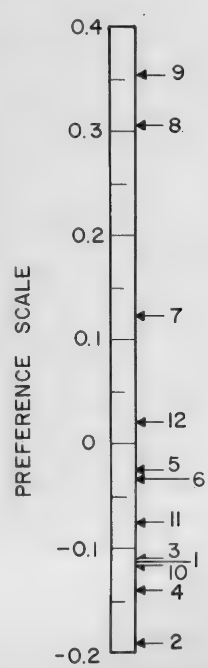


Fig. 6 Preference scale values for all 124 subjects.

artistic training were virtually identical to those of the subjects with no artistic training.

The results of the present experiment indicated that some subjects preferred patterns with a medium range of randomness, while others preferred the patterns at the extremes of randomness. The differences among all subjects were so great that preferences could not be averaged across subjects without committing a severe injustice to the data. Conventional scaling techniques, if applied to the results of the present experiment, would have produced a conclusion consistent with these past experiments, but the powerful techniques of multidimensional analysis were able to accommodate the individual preferences of each subject and show that the preferences were different enough that they could not be averaged together.

Discussion and Conclusion

Mondrian has been widely acclaimed as the "greatest Dutch painter of our time" (Bradley, 1944) and as one of the "most influential masters of painting" (Lewis, 1957). However, a computer-generated random pattern was preferred over the pattern of one of Mondrian's paintings. Furthermore, the majority of the subjects participating in the experiment were unable to correctly identify the computer-generated picture.

Both patterns were conceived by humans, although certain features of the computer-generated picture were decided by a programmed random algorithm. The computer functioned only as a medium performing its operations under the complete control of the computer program written by the programmer-artist. As stated before, the programmer-artist working with the computer produced a pattern that was preferred over the pattern of one of Mondrian's paintings. This would seem to detract from Mondrian's artist abilities. However, artistic merit is not generally accepted as something that can be determined by a jury. The experiment was designed solely to compare two patterns that differed in elements of order and randomness. It is only incidental that the more-orderly pattern was painted by Piet Mondrian while the preferred random pattern was produced with the assistance of a digital computer.

The randomness introduced by the computer was in the form of a mathematical algorithm for computing sequences of uncorrelated numbers. Thus, the "randomness" is completely deterministic, and the resulting pattern is mathematically specified in every detail. The writing of the computer program was done in an objective manner incorporating appropriate mathematical formulas. All of this indicates that no attempt was made to communicate any emotions on the part of the programmer to the final computer pattern. Therefore, the experiment compared the results of intellectual, non-emotional endeavor involving a computer with the pattern produced by a painter whose work has been characterized as expressing the emotions and mysticism of its author. The results of this experiment would seem to raise some doubts about the importance of the artist's milieu and emotional behavior in communicating through the art object. But then again, many present-day aestheticians do not subscribe to such definitions of art, and some even question whether art can be attributed any defining properties.

Since xerographic copies of a photograph of the Mondrian painting were used as stimuli in the experiment, any artistic effects due to the size or painting techniques were eliminated. The subjective comparisons hence were only on the basis of differences between the two patterns. Also, only one particular painting by Mondrian and only one particular random realization by the computer were used.

Clearly, the computer picture was more random than the Mondrian. Further programming of the computer, however, has indicated that more elaborate schemes can be used to produce a picture that even more closely resembles the Mondrian. Undoubtedly, an indistinguishable pair could finally be obtained.

The results of the computer-generated picture experiment raise questions about the special aesthetic sense frequently attributed to artists. If artistic training is requisite for classification as an artist and if aesthetic preferences for random patterns are at least a part of artistic judgment, then the results of this experiment, under these assumptions and for the carefully controlled conditions of the experiment, would seem to refute the popular assumption that artists possess a special aesthetic sense. Perhaps the aesthetic judgment of each artist is distinctly individualistic and should be valued for this individuality.

Artistic judgment has many different facets, and any number of them might be involved with judging random patterns. For this reason, the data presented in this paper, which pertain only to artistic judgments of a small subset of random patterns, can only begin to elucidate the totality of artistic judgment.

REFERENCES

BRADLEY, J. 1944. "Greatest Dutch Painter of our Time." *Knickerbocker Weekly*, Feb. 14, 3, (51), 16.
LEWIS, D. 1957. *Mondrian*. New York: George Wittenborn, Inc.
MONDRIAN, P. 1945. "Toward the True Visiton of Reality." In *Plastic Art and Pure Plastic Art*, New York; Wittenborn and Company.
SEUPHOR, M. 1962. *Abstract Painting*. New York: Harry N. Abrams, Inc.
SEUPHOR, M. 1957. *Piet Mondrian*. New York: Harry N. Abrams, Inc. ■

FINAL EXAMS —Let the Computer Write Them

Bernard Eisenberg*

At one of its meetings earlier this year, the CUNY Board of Higher Education decided that units of The City University would move to a trimester system by September 1977. Kingsborough Community College formulated a modified trimester plan to start during the Fall 1976 semester. This plan was approved by the governing body at the College and the Board. The plan calls for an academic year consisting of two 12-week semesters, one in the fall and the other in the spring, and two six-week semesters, one in the winter and the other in the summer. We usually refer to this as the 12/6/12/6 plan. Without going into further details about the plan, it means that for each academic year, at least four final examinations will have to be prepared for each of the basic courses offered in all semesters. In addition, makeup finals will also have to be produced for those students who for legitimate reasons did not take the final examinations at the scheduled time.

The Old Way.

In the Math Department, of which I am a member, the usual procedure for preparing a final examination in courses having a large number of sections, is to form a committee of five faculty members who are teaching the same course. After three or four meetings at which the committee members and others teaching the course submit possible questions and comments regarding the examination, a draft of the final is assembled and circulated to all faculty teaching the course. The draft is then modified one or two meetings later after additional comments are made, and this becomes the final examination for the course. The examination is given to the typing service and after it is typed, it's reviewed by a faculty member for typing and mathematical errors. After these errors are corrected, the examination is photo-offset and enough copies printed to accommodate all sections.

In the evening session, each of the three or four faculty teaching these courses prepares his own final and has it typed and reproduced.

This process usually starts about halfway through the semester and deadlines are given for submission of the examination to permit typing and photo-offset. If the examination is not ready for typing on time, it becomes the responsibility of the faculty members involved to type, if necessary, and reproduce in whatever manner possible, the required number of final examination copies. Frequent notices from Deans are sent to the Chairmen and faculty reminding them about the deadlines. It is not unusual to have some final examinations submitted on the last day of the semester or even on the day of the final. This is one reason some Deans and Chairmen become bald rather fast.

Computer to the Rescue.

How nice it would be to have a computer produce a final examination in about three minutes by typing a couple of words into a teletypewriter connected interactively to the computer. Well, the day of the computerized math final is here and there appears to be no reason why this couldn't be done for any course in any discipline.

The Math 05 course entitled, "Intensive Math Review," was the first course selected for the preparation of a computerized final examination. This is a remedial course required of all students who don't do well on the placement examination. There are approximately 600 to 800 students enrolled in about 30 day and evening sections of this course during each of the regular semesters and about 100 to 150 students enrolled in four to six sections during the summer session.

*Dept. of Mathematics and Computer Science, Kingsborough Community College, City University of New York.

The course covers a review of arithmetic and elementary algebra. It offers no credit and has been taught for four hours a week during the regular sessions.

Specifications.

The specifications that had to be met by the computerized final were essentially those that should be met for any final examination.

1. The questions produced had to be diverse enough each time the computer was run so that they were representative of that term's work and the curriculum covered.

2. Although some questions might have the same wording from one examination to another, the numbers in the questions had to be different for each examination.

3. The numbers used in each of the problems had to be randomly suitable. This means that the numbers selected by the computer had to be such that solutions of the problems presented would not come out to six decimal places when integers were desired, that division by zero wouldn't occur in the problems and that unusually large numbers wouldn't appear when smaller ones were desired, and vice-versa. The numbers in the problems had to reflect those used in similar classroom exercises.

4. Each time a final is to be produced, the order in which questions appear must be random; that is, it should be equally likely for any question to be first or second or third, etc.

5. Provision should be made to give students a choice from those questions selected by the computer so that any deviations in covering the course material by the large number of faculty teaching it, would still permit the student to answer a sufficient number of questions from the curriculum covered by his/her teacher.

6. The program has to be flexible enough to incorporate new material that might be introduced from term to term or to delete questions no longer relevant.

In order to meet these requirements, the last six final examinations given in Math 05 were reviewed. There appeared to be a core of common questions in both arithmetic and elementary algebra that were repeated from term to term. For example, students were asked how to add, subtract, multiply and divide whole numbers, decimals, fractions and mixed numbers. Some questions required them to convert decimals to fractions and percents and vice-versa. They had to know how to apply these conversions in real-world problems. They were required to solve linear numerical and literal equations as well as quadratic equations. They were also required to know how to factor an

Here's one final exam prepared by the computer...

```
 MATH  05 FINAL EXAM (PREPARED BY YOUR FRIENDLY CUNY COMPUTER)
PART 1
---- -
PLEASE ANSWER ANY 10   QUESTIONS OUT OF THE 12   QUESTIONS IN THIS PART.
------ ------ --- --   --------- --- -- --- --   --------- -- ---- ----
 1. CONVERT 11.0% TO AN EQUIVALENT DECIMAL AND FRACTION.
 2. GABE   SPENT 4/8 OF HIS INCOME ON HIS HOME.IF HE EARNED
       $28904. , HOW MUCH DID HE SPEND ON HIS HOME?
 3. CONVERT THE FRACTION 6/16 TO A DECIMAL AND TO A  PERCENT.
 4. ROUND: 51.3226 TO THE NEAREST HUNDREDTH.
 5. MULTIPLY:    25.980 BY  2.0
 6. SUBTRACT:   3 6/9 FROM 6 9/12
 7. COMBINE AND EXPRESS IN LOWEST TERMS:   9/7 - 1/4 + 7/6
 8. COMPUTE:   30% OF 21.6
 9. DIVIDE:   85.25 BY  7.2  (GIVE ANSWER TO NEAREST TENTH.)
10. DETERMINE WHICH OF THE FOLLOWING FRACTIONS IS THE
       LARGEST:  7/30 ,  6/29 ,  8/31  .
11. DIVIDE:   5 5/6 BY 6 3/8
12. IF  6 PENCILS COST  60 CENTS,FIND THE COST OF 19 PENCILS.


PART 2
---- -
PLEASE ANSWER ANY 10   QUESTIONS OUT OF THE 12   QUESTIONS IN THIS PART.
------ ------ --- --   --------- --- -- --- --   --------- -- ---- ----
 1. SOLVE FOR Y:   Y + 8 = 6
 2. HOW MANY CENTS ARE THERE IN X QUARTERS?
 3. COMBINE LIKE TERMS:   2X + 4Y - 2X + 3Y -  7X -  9Y

                          3
 4. FACTOR COMPLETELY:   4X  -  16X

                          9 4      2 3     4 2
 5. FACTOR COMPLETELY:   9R S + 27R S - 36R S    .
 6. FIND THE VALUE OF W IF W = (6X - 1Y)/(7Z - 8U) AND
       X = 1, Y = 2, Z = 3.3, AND U = 1
 7. SOLVE FOR T:   9T/1 = 5/15
 8. MULTIPLY:   (6V - 8)(6V + 8)
 9. SIMPLIFY:   (2X - 12)/(X SQUARED - 36)
10. SOLVE FOR G:   AG - 9 = BG +  7
11. SOLVE FOR Y:   Y SQUARED -  9 = 0

                                  2 9     5 8
12. SIMPLIFY THE FRACTION:   12A B C/24A B C


 PART 3
 ---- -
 PLEASE ANSWER ANY 4    QUESTIONS OUT OF THE 7    QUESTIONS IN THIS PART
 ------ ------ --- --   --------- --- -- --- --   --------- -- ---- ----
  1. SOLVE THE FOLLOWING 2 EQUATIONS ALGEBRAICALLY FOR X AND Y:
         5X + 2Y =   4
         7X - 4Y = -42

  2. SIMPLIFY THE FOLLOWING FRACTION:
         1       1
         -   +   -
         2       7
         ---------
         3       2
         -   -   -
         4       3

  3. THE LENGTH OF A RECTANGULAR GARDEN IS 39 LONGER THAN ITS
         WIDTH. IF 650 FT. OF FENCING ARE NEEDED TO ENCLOSE THE
         GARDEN, WHAT ARE ITS LENGTH AND WIDTH?

  4. IF 35 POUNDS OF FERTILIZER WILL COVER  735 QUARE FEET OF LAWN,
         HOW MANY POUNDS ARE NEEDED TO COVER A RECTANGULAR
         LAWN  49 FT. BY  42 FT.?

  5. JULIO  HAS 30 STAMPS. SOME ARE  3 CENT STAMPS AND THE OTHERS
         ARE  6 CENT STAMPS.IF THE TOTAL VALUE OF THE STAMPS IS
         $1.50, HOW MANY STAMPS OF EACH KIND DOES JULIO  HAVE?

  6. TWO TRAINS THAT ARE 1120 MILES APART, TRAVEL TOWARDS EACH
         OTHER. ONE TRAVELS AT  69 MI./HR. AND THE OTHER TRAVELS
         AT  91 MI./HR. HOW LONG WILL IT BE BEFORE THEY MEET?

  7. SOLVE GRAPHICALLY FOR X AND Y:
        X - 2Y =   3
       5X + 1Y =  37
```

...and here's another generated by the same program.

```
 MATH  05 FINAL EXAM (PREPARED BY YOUR FRIENDLY CUNY COMPUTER)
PART 1
---- -
PLEASE ANSWER ANY 10   QUESTIONS OUT OF THE 12   QUESTIONS IN THIS PART.
------ ------ --- --   --------- --- -- --- --   --------- -- ---- ----
 1. IF 11 PENCILS COST  77 CENTS,FIND THE COST OF 24 PENCILS.
 2. SONNY  SPENT 4/9 OF HIS INCOME ON HIS HOME.IF HE EARNED
       $29133. , HOW MUCH DID HE SPEND ON HIS HOME?
 3. DIVIDE:    4.46 BY  7.0  (GIVE ANSWER TO NEAREST TENTH.)
 4. CONVERT 16.7% TO AN EQUIVALENT DECIMAL AND FRACTION.
 5. DIVIDE:   8 2/6 BY 8 7/8
 6. COMPUTE:   36% OF 32.4
 7. FIND THE AREA OF A TRIANGLE IF ONE SIDE IS  7 FT. LONG AND
       THE ALTITUDE UPON THAT SIDE IS 12 FT.
 8. ROUND: 67.7431 TO THE NEAREST HUNDREDTH.
 9. MULTIPLY:    25.790 BY  4.3
10. AN ARTICLE SELLS FOR  $20.12. A 6% SALES TAX IS ADDED.
       WHAT IS THE TOTAL PAID FOR THE ARTICLE?
11. COMBINE:   146.428 -  5.73 +9.7
12. SUBTRACT:   6 5/9 FROM 9 1/12


PART 2
---- -
PLEASE ANSWER ANY 10   QUESTIONS OUT OF THE 12   QUESTIONS IN THIS PART.
------ ------ --- --   --------- --- -- --- --   --------- -- ---- ----
 1. HOW MANY CENTS ARE THERE IN X QUARTERS?
 2. SOLVE FOR G:    AG - 6 = BG + 12     .
 3. FACTOR COMPLETELY:    B SQUARED - 2B -  8
 4. WRITE AN EQUATION FOR THE FOLLOWING WORD PROBLEM
       BUT DO NOT SOLVE.  9 TIMES A NUMBER IS 4 MORE
       THAN 4 TIMES THE NUMBER. FIND THE NUMBER.
 5. SOLVE FOR Y:    Y + 6 < 1
 6. SOLVE FOR Y:    Y SQUARED - 36 = 0

                                     4 9     7 2
 7. SIMPLIFY THE FRACTION:    36A B C/24A B C  .
 8. COMBINE INTO A SINGLE FRACTION:    8/(9X) - 9/(45X)
 9. COMBINE LIKE TERMS:    1X + 5Y - 3X + 8Y -  5X - 12Y

                              8 7      2 3      4 2
10. FACTOR COMPLETELY:    3R S +  9R S - 12R S     .
11. MULTIPLY:    (3V - 7)(3V + 7)
12. FIND THE PERIMETER OF A TRIANGLE WHOSE SIDES ARE
       (3X + (-4Y)),(-2X + 8Y), AND ((-9Y) - 7X)


PART 3
---- -
PLEASE ANSWER ANY 4    QUESTIONS OUT OF THE 7    QUESTIONS IN THIS PART.
------ ------ --- --   --------- --- -- --- --   --------- -- ---- ----
 1. IF 54 POUNDS OF FERTILIZER WILL COVER 1782 SQUARE FEET OF LAWN,
       HOW MANY POUNDS ARE NEEDED TO COVER A RECTANGULAR
       LAWN  77 FT. BY  84 FT.?

 2. SIMPLIFY THE FOLLOWING FRACTION:
       1       1
       -   +   -
       4       9
       ---------
       3       2
       -   -   -
       4       3

 3. SOLVE THE FOLLOWING 2 EQUATIONS ALGEBRAICALLY FOR X AND Y:
       7X + 6Y =  14
       7X - 1Y = -35

 4. THE LENGTH OF A RECTANGULAR GARDEN IS 21 LONGER THAN ITS
       WIDTH. IF 350 FT. OF FENCING ARE NEEDED TO ENCLOSE THE
       GARDEN, WHAT ARE ITS LENGTH AND WIDTH?

 5. MARCEL HAS 50 STAMPS. SOME ARE  4 CENT STAMPS AND THE OTHERS
       ARE  7 CENT STAMPS.IF THE TOTAL VALUE OF THE STAMPS IS
       $3.20, HOW MANY STAMPS OF EACH KIND DOES MARCEL HAVE?

 6. A COLLECTION OF 54 COINS CONSISTS OF DIMES AND QUARTERS
       AND HAS A VALUE OF $10.80. HOW MANY OF EACH ARE THERE?

 7. ONE NUMBER IS 5 MORE THAN 7 TIMES A SECOND NUMBER.
       THEIR TOTAL IS 29. FIND THE NUMBERS.
```

algebraic expression, how to apply this in the solution of equations, and how to solve a system of two linear equations algebraically and graphically. They had to perform the arithmetic operations on algebraic fractions and expressions and, finally, they had to know how to solve a variety of word problems algebraically.

Program Development.

To develop a computer program to simulate these exams, it was decided to divide the examination into three parts. The first would cover arithmetic, the second would cover elementary algebra, and the third would contain word problems, simplification of a complex fraction and problems requiring the solution of two simultaneous equations algebraically and graphically. A bank of 18 arithmetic questions was prepared for Part 1. A bank of 18 elementary algebra questions was developed for Part 2 and a bank of 10 questions was selected for Part 3.

Here's how the program works. Let's assume that the faculty have decided that the students should answer 10 out of 13 questions in Part 1, 10 out of 13 in Part 2 and 4 out of 7 in Part 3. Let's further assume that such a decision by the faculty will not take more than a day. The latter assumption may be somewhat risky, but not unreasonable. On the first day of classes or during the first week, the program (developed in BASIC) is run using the aforementioned selectivity numbers and within three minutes, the final examination produced is the final examination that the student gets. This examination will contain randomly-selected questions from each bank, they will be randomly presented in the examination, and each program run will contain different and suitable numbers for each of the questions. (Two sample copies produced in a six-minute period are shown.) The computer printout of the examination is given to the reproduction department for photo-offset and as far as the faculty are concerned that final is finished. No further typing is needed.

Benefits.

Faculty don't have to spend long hours meeting and bothering each other to meet deadlines. The time saved by the faculty can be devoted to their students and teaching rather than to the chores of preparing the final. To improve this program still further, one could make it a multiple-choice-type examination which could be machine-graded. This however, would do the student a disservice, in math courses at least, since we're interested in the work done and method used to support the answers attained. The reasoning and thinking used to solve a problem are more important than the answers.

Time Saved Using Computerized Finals

A reasonable estimate of time saved in a semester by using computerized finals for Math 05 is as follows:

Faculty time for final exams: 50 hours/semester — day session
15 hours/semester — eve. session

Typing time for final: 2 hours — day session
6 hours — eve. session

Faculty time for 3 classroom exams: 72 (assuming 2/3 of faculty participate)

With savings of this magnitude, each regular semester and probably a third to half as much for each six-week winter and summer session, it would appear economically feasible to computerize all final and classroom exams for basic courses in each discipline. The larger the number of multiple sections for a course, the greater will be the faculty time saved.

Nevertheless, a second program has been produced which not only presents questions and suitable numbers randomly, but also gives the answers to each of the questions. This also saves some faculty time when they prepare to grade the examinations. For the benefit of bilingual students, a third program will produce a random final examination in Spanish, with answers as well.

Other Savings.

There are other savings and benefits from the use of computer-prepared random final examinations that will accrue to faculty and students as well. During the last two weeks of each semester it is customary for faculty to prepare a review sheet that serves as a guide for students in preparing for the final. The review sheet usually highlights those topics the student will be responsibile for and will most likely contain a list of problems he/she should know how to solve. The faculty members no longer have to do this. Once again, on the first day of the term, instead of during the last two weeks, the complete bank of questions in all three parts of the examination can be run off by the computer in about three minutes and again the questions are randomly positioned in each part and the numbers used are suitable and randomly selected by the computer. In fact, in those questions where a person's name is mentioned, the name is also randomly selected. The complete set of questions may be given to the students and they can be told that the major portion of the final examination will be selected from this set. The goals for the course are thereby set at the first session. As an alternative to giving the students the entire bank of questions, a separate bank of 30 mixed problems in arithmetic and algebra is available and can be used.

It frequently happens that with large groups of students taking the same course, the final exam is usually administered in large lecture halls with students occupying alternate seats. By using three different computerized finals for the same course, all the seats in the lecture halls may be used and final exam number 1 may be given to those students occupying columns 1, 4, 7, 10, etc., exam number 2 may be given to those in columns 2, 5, 8, 11, etc., and exam 3 may be given to those in columns 3, 6, 9, 12 etc. As a consequence, fewer proctors will be required to administer the exam and student cooperation on the exam is minimal.

Improving the Quality.

To improve the quality of the exam, the bank of questions in each part will be increased by at least 50 percent during the current semester so that there will be at least 27 questions in each of Parts 1 and 2 and at least 15 in Part 3. Any other suggestions from faculty and students who use these exams this semester will be incorporated in the next master program. Other future plans call for computerizing the three exams given during the semester. Since an exam is usually given after several topics are covered, the bank of questions from which each of the classroom exam questions is selected will be solely from these topics. Starting with the winter session, all classroom and final exams in Math 05 may be computerized with and without answers and bilingually in Spanish. All makeup exams in Math 05 which have to be prepared, can be done so in less than three minutes. Although not all faculty (about 13) teaching Math 05 may wish to join this project for one reason or another, the initial reaction appears to be that most, if not all, will. All of the evening-session final exams can also be run on the computer and a different one may be given to each evening session instructor.

Future projects will consist of the development of similarly computerized exams for such multiple-sectioned courses in Math and then in other disciplines. ■

Topics in Logic

Part 2: Formal Grammars and Languages

John Lees

The preceding article in this loosely-connected series gave an introduction to computer languages. Now we're going to take a closer look at one of the primary tools for understanding and manipulating languages, the formal grammar.

Underlying every language; natural or artificial, is a grammar, or set of rules for producing the language. In English, we say that a sentence is correct if it is produced according to the rules of the English grammar. "The is paper white" is not a correct sentence because it cannot be produced according to the rules of the English grammar. "GO TO TOWN" is a correct English sentence; it is also a correct PL/I sentence, but not a correct Basic sentence. It all depends on the grammar.

It could be noted in passing that "The door walks with green melodies" is a correct English sentence. It doesn't *mean* anything, but it is grammatically correct. Grammar (syntax) and meaning (semantics) are not necessarily connected. However, it is generally agreed that it is uncool in a computer language to allow syntactically correct sentences without meanings. But do keep in mind that grammar and meaning are two entirely different things, perhaps especially when dealing with computers.

Let's introduce a distinction. Not all grammars are *formal* grammars. A formal grammar is one which we can completely specify in a particular way using a standardized notation. Very few of the languages we use can actually be defined with a formal grammar. The "natural" languages, such as English and French, cannot be formally defined. That is one reason why we don't have machines which can understand English or translate between natural languages.

You are all familiar with one very simple language, that of numerals. Things like "123" or "02145". Being a simple language, it has a simple grammar:

$G_1 = (V_N, V_T, P, S)$

$V_N = \{$S, NUMERAL, DIGIT$\}$

$V_T = \{0,1,2,3,4,5,6,7,8,9\}$

P = { S: : = NUMERAL
NUMERAL : : = DIGIT
NUMERAL : : = DIGIT NUMERAL
DIGIT : : = 0 |1 |2 |3 |4 |5 |6 |7 |8 |9 }

Looks rather messy, doesn't it? Well, it isn't. Most of that is just formal notation. The first line is the definition of this grammar, G_1. V_N, V_T, and P are sets, and S is an element of set V_N called the start symbol. V_N is the set of non-terminal symbols, that is, symbols used in the grammar which don't end up in the language. V_T is the set of terminal symbols, those which do end up in the language, and P is the set of rules, or productions, which define how the terminal symbols may be combined to produce sentences of the language. In the productions, ": : =" means "produces" or "is replaced by," and the "|" means "or". The language, L_1, defined by the grammar, G_1, is the set of all possible strings, or sentences, produced by the grammar. The language L_1, like most interesting languages, is infinite.

Now how does this grammar work? Let's see if we can write the set of rules which produce the sentence "02145". We begin with the production having the start symbol as a left-hand side.

S : : = NUMERAL

Now we replace a non-terminal on the right-hand side with the right-hand side of a production of which the non-terminal is the left-hand side. Better read that sentence again. We want a multi-digit numeral, so we'll make the following substitution of "DIGIT NUMERAL" for "NUMERAL". We can do this because NUMERAL, the right-hand side of the production S : : = NUMERAL, appears as the left-hand side of the production NUMERAL : : = DIGIT NUMERAL.

S : : = DIGIT NUMERAL

At this point we can subsittute for DIGIT a terminal, specifically 0, giving:

S : : = 0 NUMERAL

Then we substitute for NUMERAL AGAIN:

S : : = 0 DIGIT NUMERAL

For DIGIT:

S : : = 02 NUMERAL

Keep on like that until we have:

S : : = 0214 NUMERAL

This time we make a different substitution for NUMERAL, since we don't want the numeral to grow any larger:

S : : = 0214 DIGIT

And a final substitution for DIGIT:

S : : = 02145

And voila! We have produced a sentence in the language of numerals using the grammar G_1. What we just did, incidently, is what is known as "top-down parsing." Bottom-up parsing would reverse the process, beginning with the numeral and making substitutions of left-hand sides for right-hand sides until we ended up with the start symbol on the left-hand side; that is S : : = NUMERAL. Both techniques of parsing have the same purpose: to determine if a given sentence is in a certain language. You parse a sentence to see if it is syntactically correct.

The grammar we just looked at has one little deficiency. Normally we don't want leading zeros on numerals. A small modification to the grammar yields a new grammar for a new language which doesn't have leading zeros. Listing only the productions this time:

S : : =	NUMERAL
NUMERAL : : =	DIGIT
NUMERAL : : =	NZDIGIT STRING
STRING : : =	DIGIT STRING
STRING : : =	DIGIT
DIGIT : : =	0 |1 |2 |3 |4 |5 |6 |7 |8 |9
NZDIGIT : : =	1 |2 |3 |4 |5 |6 |7 |8 |9

Yes, all that added complication just to forbid leading zeros. You begin to see why grammars are not the solution to the world's ills. Try this new grammar out and see if you can produce the strings "123", "20", "5", "0". After doing so, you'll see why all those productions are necessary.

What happens if we try to parse the sentence "02145" with G_2, our new grammar? Doing it bottom-up:

02145

0214 DIGIT

0214 STRING

021 DIGIT STRING

021 STRING

02 DIGIT STRING

02 STRING

0 DIGIT STRING

0 STRING

At this point we're stuck. We can replace "0 STRING" with "DIGIT STRING" and that with "STRING", but there is no substitution of "NUMERAL" for "STRING" and hence no way to get to the start symbol, S. The string "02145" is not in the language defined by G_2.

Nifty, huh? If you're clever you can even write a program to parse a string according to G_2 and print out a message such as INVALID CONSTANT if it can't. It is but a large and difficult step from this small grammar and such a simple program, to one which can parse the sentence GO TO TOWN and reply SYNTAX ERROR because the sentence is not in the Basic language.

We have just scratched the surface of the topic of grammars and languages. There are different types of grammars, and techniques for converting one type into another and reasons why you'd rather have another type, and all kinds of theorems and algorithms for this and that. A language can have more than one grammar. There exist enormous programs which will take as input certain types of grammars and produce as output a complete compiler for the language defined by the grammar. The area of formal languages is one of the major subject areas of computer science.

Not that the Basic you use on your friendly neighborhood computer was produced by a compiler generator. The Basic language is not that well-behaved. It's easier to write a compiler by hand; in most cases, although you certainly use as much of a grammar as you can figure out to help you. Attempting to formalize the grammar for a language, or just for each statement type, is a great way to catch disasters such as ambiguity. There are, by the way, grammars for most of the tiny languages floating around and you sometimes run across a language manual which uses the notation of grammars (BNF, its called) to set out the syntax of statements.

Before we close, let's consider ambiguity, because it is interesting and is one of the main reasons why English doesn't have a nice grammar. Formally, a sentence is ambiguous if there is more than one derivation for it, more than one distinct set of productions which result in that sentence. Consider a very simplified English grammar:

SENTENCE : : =	SUBJECT PREDICATE | PREDICATE SUBJECT
SUBJECT : : =	NOUN
PREDICATE : : =	VERB

Consider a very simple English sentence "Time flies." Now parse it:

Time flies

Time VERB

NOUN VERB

NOUN PREDICATE

SUBJECT PREDICATE

SENTENCE

Fine. "Time flies" is a syntactically correct English sentence. But what's wrong with:

Time flies

Time NOUN

VERB NOUN

PREDICATE NOUN

PREDICATE SUBJECT

SENTENCE

Nothing is wrong with it! Either way, "Time flies" is an English sentence. Voila! We have just shown that our simplified English grammar is ambiguous. Complicating the grammar won't help at all. As you well know from a lifetime of experience, English is what is known as an inherently ambiguous language. Spoken English is even worse. The problem with an ambiguous language comes when you try to attach a meaning to a sentence. If the language is unambiguous, you can have the nice, neat situation of one sentence, one meaning. Not so in English!

People who write compilers for computer languages long for unambiguous languages. They would be quite happy if once they parsed a sentence they could attach a single, definite meaning to it. But life is not that simple.

Take Fortran. (Please!) A harmless statement such as ALPHA = WEIGHT(X). Is WEIGHT an array, or a function? The grammar won't tell you. Writing a compiler can be a very, very tricky business. Compiler writers cheat a lot.

In spite of suffering from somewhat ambiguous syntax, once you settle on a particular derivation for a sentence in a computer language, at least the meaning isn't ambiguous. Alas for us poor English-speaking creatures! There are English sentences that parse quite nicely, but are as ambiguous as all get out. The woe caused over the ages by the semantic ambiguity of that simple and common English sentence "I love you" is legion ... and poetry. Predicate, subject, direct object. Ah, but there is more to language than syntax! Perhaps next time we'll delve into semantics. ■

puzzles & problems

New Life for Nim!

by B. M. Rothbart, London

Nim is perhaps the most popular game as a programming exercise in beginning computer science courses. However, as far as playing the game for the initiated player, it is no more than a test of adding binary numbers in one's head.

However, a small change produces a two-dimensional Nim which resists attempts at a definite analysis. Objects are placed in small groups forming a two-dimensional array as follows:

```
2 7 6 3
1 2 0 5
4 3 9 1
```

The two players then take turns to remove matches with the sole limitation that the matches removed on any turn must be from the same column or row. As in conventional Nim, the player who takes the last match can be either the winner or loser depending on the agreement reached before the game started.

Some trivial wins and losses are readily spotted, but the charm lies in the way in which even experienced players are unable to play by rote. Is there a foolproof method for playing? We leave that for readers to find out.

Different Numbers

by
Eve R. Wirth

In this puzzle; first you must supply the 12 missing numbers and then add them up. This will give you a year in which one of the underlying principles of the computer was invented. (Hint: it was a mathematical principle.) What was the year, the principle, and who was the person?

1._____ Nights" ak/a The Arabian Nights
2._____ degrees in a circle
3._____ original colonies
4._____ signs in the Zodiac
5._____ square inches = 1 square foot
6._____ th Amendment (Women's Suffrage)
7._____ Years War (Anglo-French wars)
8._____ Heinz Varieties
9._____ Degrees (boiling point of water Fahrenheit thermometer)
10._____ feet = 1 fathom
11._____ R's (basics of education)
12._____ good turn deserves another.
_____ Total

Answer to "Different Numbers"

1.	1001	5.	144	9.	212
2.	360	6.	19	10.	6
3.	13	7.	100	11.	3
4.	12	8.	57	12.	1

The total is 1928 and if you have a good mathematics history book you will discover that Vannevar Bush devised differential analysis in that year. You will be forgiven if you could not find Bush in your book (he is not widely recognized), however, you might have found that Fleming discovered Penicillin that year and Pressey built the first teaching machine.

Simple (Crypt) Arithmetic

```
  N I N E
- F O U R
  -------
  F I V E
```

```
    O N E
    T W O
+ F I V E
---------
E I G H T
```

```
    O N E
    T W O
+ F O U R
---------
S E V E N
```

```
  S E V E N
  S E V E N
+     S I X
-----------
T W E N T Y
```

```
  F I V E
- F O U R
---------
    O N E
+   O N E
---------
    T W O
```

```
F O R T Y
    T E N
+   T E N
---------
S I X T Y
```

MULTIPLY

```
    G E T
      O N
    -----
    R O N
  G E T
  -------
  G R A N
```

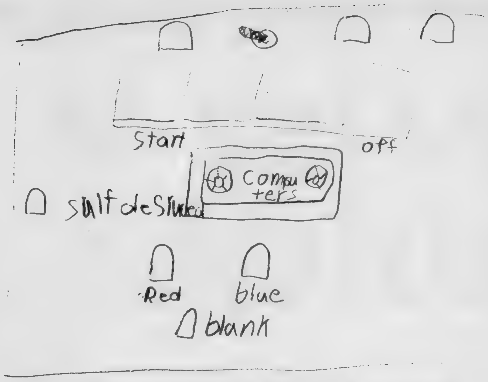
DIVISION

```
        M A N
     --------
B E  )A B L E
      M N
      ---
        L L
        A T
        ---
          H E
          H E
          ---
```

"This year's social studies project," I ingeniously announced to the moppets in my elementary class, "is to learn about the computer industry." I proceeded to produce films, books, magazine articles, lectures and the obligatory field trip.

When the term had come to its close, they demonstrated the wealth of their new knowledge in test papers, reports, and homework assignments.

If you are overworked, discouraged or tired, have no fear, the next generation is almost ready, as you will quickly ascertain from these quotes from their writings.



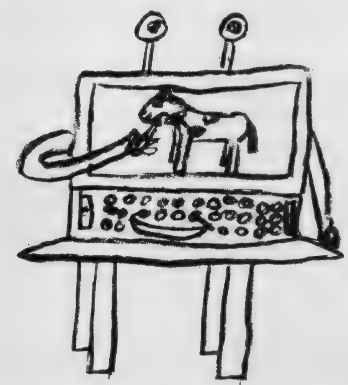
"Take a good long look at a computer. Does it have input, output, a bit of binary? No, you say to all these questions? Then you are not taking a good long look at a computer."

Out of the Mouths of Babes

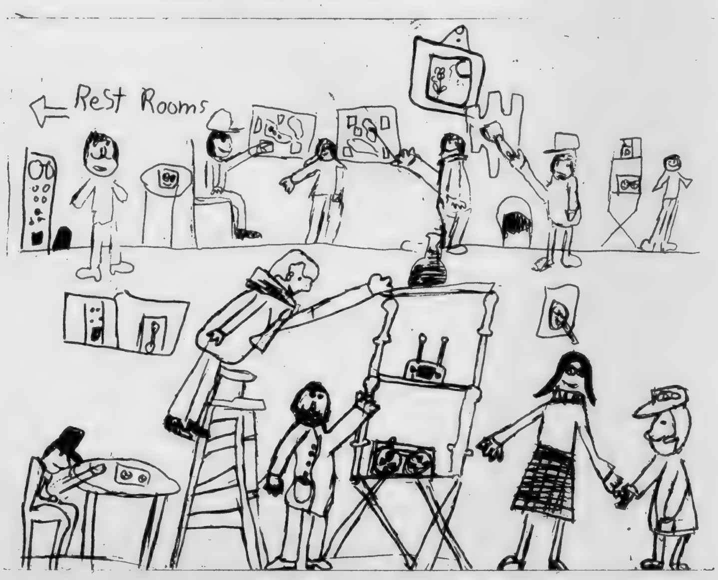
Eve R. Wirth

"Just yesterday when I read my libury book I knew real good what computers axaly do for us, but today it's a different colored horse story."

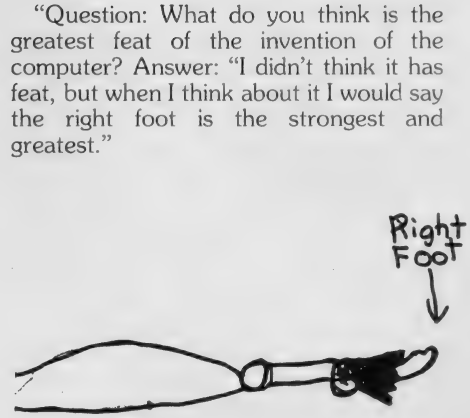
"Girl computer workers have to make real certain all the holes are in the right spots, because if not then how will the computerman get in them."



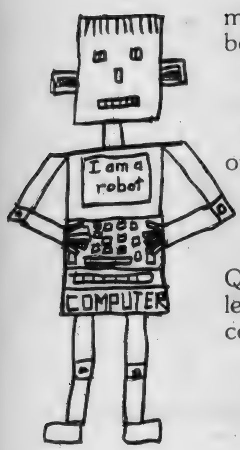
"Lots of people are working in the hustling bustling business of computers. There could easily be 1000 of them doing this. Maybe 5000 doing this. Might even be a million. I can't be too for sure because it takes just about all of my knowing to even know that lots of people are working in the hustling bustling business of computers."



"Question: What do you think is the greatest feat of the invention of the computer? Answer: "I didn't think it has feat, but when I think about it I would say the right foot is the strongest and greatest."



"In the pre-me times of history one day a guy decided to make a machine that could do stuff faster. He thought it was high time for action, so here's what he did. He put alot of holes in it and lots of buttons and stuff, and when you pressed the buttons zoom-voom-boom you got your answer and then everybody yelled with their deep throats woopee, yippee and maybe even sock-it-to-me."



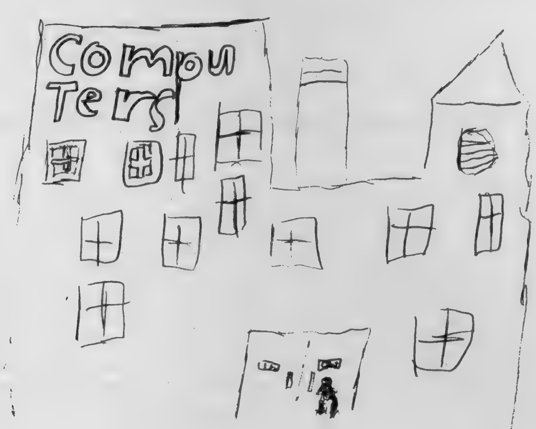
"Remembering eggacly what 'binary' means is something that is forever going to be on my mind."

"What a 'bit' is has a very short memory on my end."

Question: How long are people in school to learn about computers? Answer: "They could be anywhere from 5 feet and up."

"A bit of blarney is computer talk for all practical purposes invented by the Irish."

"I was so glad in my body to know that someday I would go to school to find out how to be a computer programmer. I had so many glad tickles in my stomach about it. Then with a sudden finding out that I also wanted to be a pilot, all my glad tickles went down my throat upside down and with a lump coming of sadness it was all over me being a computer programmer."



Question: How long have computers been in existence?
Answer: "Since the beginning of time and maybe even longer than this."

"The very first modern computer was built in the dark ages of 1930, in either the A.D. or V.D. times of history."

"If you like to fool around with figures alot then become a design engineer. My Uncle Henry is one, and he fools around alot with figures."

"A lady computer operator and a man computer operator are the same, only just the opposite in the you know where places."

"They are producing more and more people to work on computors anally."

"From now on, after learning all about computers, I'm going to think wonderful happy-that-you-made-it-so thoughts with a smile in my heart."

Aren't you? ■

Illustrations by Detta Ahl and Robert Green IV

Great Gifts From Creative Computing

Tired of returning to your local gift shop and hardware store to browse among china, baskets, appliances and tools for a present appropriate to that special person?

Instead, this year, share your interests. Choose a gift from Creative Computing — select from outrageous fun to topical information.

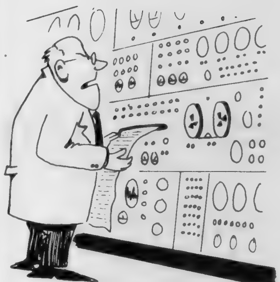
Bridge the gap. Microcomputers have such a potential for communication wouldn't it be a shame if we let ourselves become isolated from our friends and family because of our interest in absorbing them. Instead, give a subscription to *Creative Computing,* or a game, or book, or t-shirt, that say "join me in this interest, laugh and learn along with me."

Just look at all the neat stuff you can give (or ask to get)!

Give *Creative Computing* to a friend for only pennies a day!

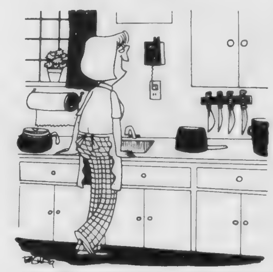
The best gift of all is a magazine delivered every other month as a reminder of your thoughtfulness. *Creative Computing* offers you the means to get the most from your computer. It includes thoroughly documented programs with complete listings and sample runs — games, CAI, artificial intelligence — music and speech synthesis, investment analysis—programming techniques—in depth book reviews, hardware and software evaluations to help you choose what's best for you, and much more.

Creative Computing will be a gift you'll be thanked for, again and again. Each subscription you send will be announced as a gift from you by a computer generated art card.



"OH YEAH!"

WELL, LET'S SEE IF THE FRONT OFFICE THINKS THIS ONE IS SIMPLE ENOUGH TO OPERATE!"



"COMPUTER, COMPUTER ON MY WALL WHO'S THE FAIREST OF THEM ALL?"

The Gift of Laughter

Is there a better way to be clued into the phenomena of computers than through computer age humorists? What is the effect these new machines are having on the way we think and relate? Laugh yourself silly to the answers as you discover the hidden wisdom and unforgettable hilarity in

The Colossal Computer Cartoon Book

If you are ready to share your passion about computers, start by giving this bright, zany collection of the wittiest cartoons ever. You get 128 pages 8½x11 inches of fun and frolic for only $4.95. It's an irresistible introduction for the novice and a startling look at the familiar for the initiated.

SET OF 3 BINARY DICE

Three binary dice, one red, green, and blue in zip-lock bag. Each die has 3 ones and 3 zeroes. Use one to simulate the flipping of a coin, with two you can count to four, just think of the possibilities with three! The only dice that let you roll zero! Wow! Set of 3, $1.25 postpaid. [3G]

COMPUTER RAGE is a fascinating new board game based on a large-scale multi-processing computer system. The object is to move your three programs from input to output. Moves are determined by the roll of 3 binary dice. Hazards include priority interrupts, program bugs, decision symbols which alter your path, power failures, and restricted use input and output channels. Notes are included for using the game in school. Ages 10-adult; 2-4 players. **COMPUTER RAGE** comes with a colorful board, 12 program playing pieces, 3 binary dice, 38 interrupt cards. Orders must be prepaid. Only $8.95 postpaid ($10 outside of USA). Creative Computing, P.O. Box 789-M, Morristown, NJ 07960, ATTN: Cindy

Place
Stamp
Here

creative computing

Box 789-M
Morristown, N.J. 07960

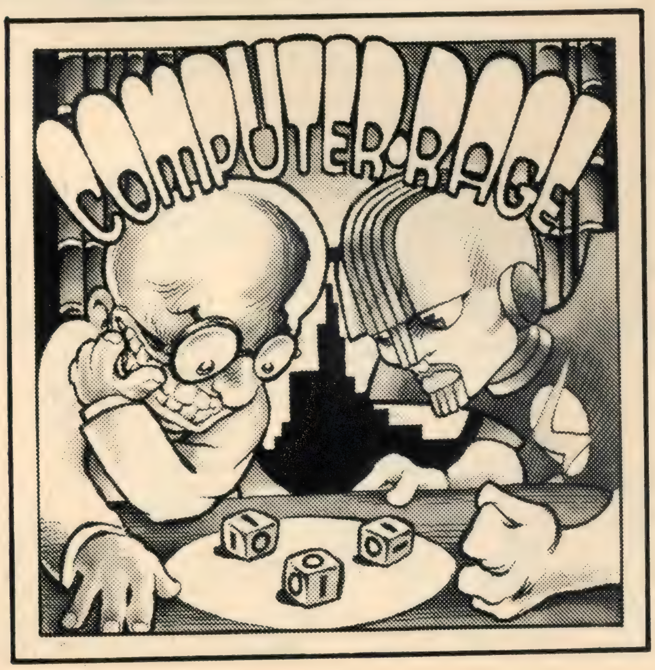
more.

Creative Computing will be a gift you'll be thanked for, again and again. Each subscription you send will be announced as a gift from you by a computer generated art card.

think and relate? Laugh yourself silly to the answers as you discover the hidden wisdom and unforgettable hilarity in

The Colossal Computer Cartoon Book

If you are ready to share your passion about computers, start by giving this bright, zany collection of the wittiest cartoons ever. You get 128 pages 8½x11 inches of fun and frolic for only $4.95. It's an irresistible introduction for the novice and a startling look at the familiar for the initiated.



SET OF 3 BINARY DICE

Three binary dice, one red, green, and blue in zip-lock bag. Each die has 3 ones and 3 zeroes. Use one to simulate the flipping of a coin, with two you can count to four, just think of the possibilities with three! The only dice that let you roll zero! Wow! Set of 3, $1.25 postpaid. [3G]

COMPUTER RAGE is a fascinating new board game based on a large-scale multi-processing computer system. The object is to move your three programs from input to output. Moves are determined by the roll of 3 binary dice. Hazards include priority interrupts, program bugs, decision symbols which alter your path, power failures, and restricted use input and output channels. Notes are included for using the game in school. Ages 10-adult; 2-4 players. **COMPUTER RAGE** comes with a colorful board, 12 program playing pieces, 3 binary dice, 38 interrupt cards. Orders must be prepaid. Only $8.95 postpaid ($10 outside of USA). Creative Computing, P.O. Box 789-M, Morristown, NJ 07960, ATTN: Cindy

creative computing

[Want faster service – call toll free 800-631-8112 In NJ 201-540-0445]

BOOKS AND MERCHANDISE

☐ Gift ☐ Send to me

[Gifts cannot be gift wrapped but a card with your name will be sent with each order]

Quan	Cat	Descriptions	Price

Books shipping charge $1.00 USA, $2.00 Foreign ______

NJ Residents add 5% sales tax ______

TOTAL (magazines and books) ______

Order Form
Subscriptions

☐ New ☐ Renewal ☐ Gift

TYPE OF SUBSCRIPTION		U.S.	Foreign Surface	Foreign Air
Individual	1-yr.	☐ $ 8	☐ $ 12	☐ $ 20
	2-yr.	☐ 15	☐ 23	☐ 39
	3-yr.	☐ 21	☐ 33	☐ 57
	Lifetime	☐ 300	☐ 400	☐ 600

YOUR NAME AND ADDRESS:

Name ______

Address ______

City ______ State ______ Zip ______

NAME TO APPEAR ON GIFT CARD:

______

SEND GIFT SUBSCRIPTION TO:

Name ______

Address ______

City ______ State ______ Zip ______

PAYMENT INFORMATION

☐ Cash, check or M.O. enclosed

☐ Visa/BankAmericard } Card no. ______

☐ Master Charge } Exp. ______

☐ Please bill me ($1.00 billing fee will be added)

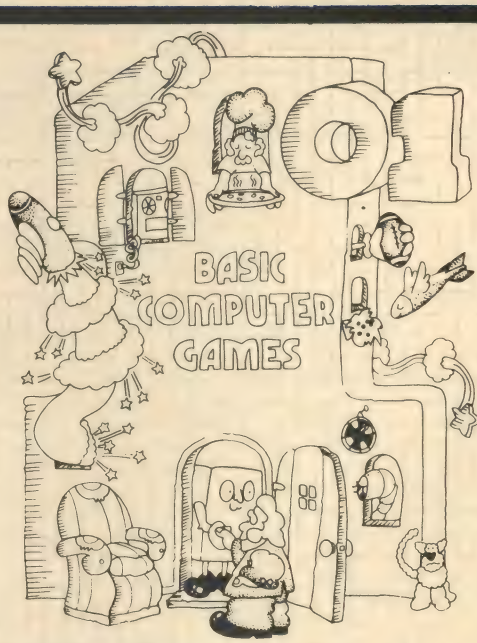
Book orders from individuals must be prepaid.

Helen P.O. Box 789-M, Morristown, N.J. 07960.

THE BEST OF BYTE — VOL. 1

The Best of Byte - Volume 1 is a 384-page blockbuster of a book which contains the majority of material from the first 12 issues of *Byte* magazine. 146 pages are devoted to "Hardware" and are cram full of how-to articles on everything from TV displays to joysticks to cassette interfaces. The section on computer kits describes building 7 major kits. But hardware without software might as well be a boat anchor, so there are 125 pages of "Software and Applications" ranging from on-line debuggers to games to a complete small business accounting system. A section on "Theory" examines the how and why behind the circuits and programs, and a final section "Opinion" looks at where this explosive new hobby is heading.

The Best of Byte - Volume 1 is edited by Carl Helmers and David Ahl and published by Creative Computing Press. Price in the US is $11.95 plus $1.00 shipping and handling ($12.95 total); foreign orders add $1.00 ($13.95 total). Orders from individuals must be prepaid. Creative Computing Press, Attn: Helen, P.O. Box 789-M, Morristown, NJ 07960. Allow 8 weeks for delivery.



101 BASIC Computer Games is the most popular book of computer games in the world. Every program in the book has been thoroughly tested and appears with a complete listing, sample run, and descriptive write-up. All you need add is a BASIC-speaking computer and you're set to go.

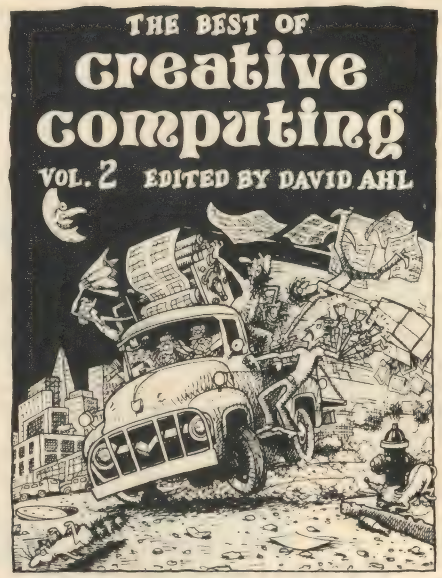
101 BASIC Computer Games. Edited by David H. Ahl. 248 pages. 8½x11 paperbound. $7.50 plus 75¢ postage and handling ($8.25 total) from Creative Computing, P.O. Box 789-M, Morristown, NJ 07960.

The diversity in *The Best of Creative Computing — Volume 1* can only be described as staggering. The book contains 328 pages of articles and fiction about computers, games that you can play with computers and calculators, hilarious cartoons, vivid graphics and comprehensive book reviews.

Authors range from Isaac Asimov to Sen. John Tunney of California; from Marian Goldeen, an eighth-grader in Palo Alto to Erik McWilliams of the National Science Foundation; and from Dr. Sema Marks of CUNY to Peter Payack, a small press poet. In all, over 170 authors are represented in over 200 individual articles, learning activities, games, reviews and stories.

This 328-page book has 108 pages of articles on computers in education, CAI, programming, and the computer impact on society; 10 pages of fiction and poetry including a fascinating story by Isaac Asimov about all the computers on earth linking up after a nuclear war to support the few remaining survivors; 15 pages of "Foolishness" including a cute cartoon piece - called "Why We're Losing Our War Against Computers"; 26 pages on "People, Places, and Things" including the popular feature "The Compleat Computer Catalogue" which gives capsule reviews and lists sources for all kinds of computer-related goodies; 79 pages of learning activities, problems and puzzles; 29 pages continuing 18 computer games including a fantastic extended version of the single most popular computer game — Super Star Trek; and 32 pages of in-depth book and game reviews including Steve Gray's definitive review of 34 books on the Basic language.

The Best of Creative Computing - Volume 1 is available by mail for $8.95 plus 75¢ postage from Creative Computing Press, Attn: Helen P.O. Box 789-M, Morristown, N.J. 07960.



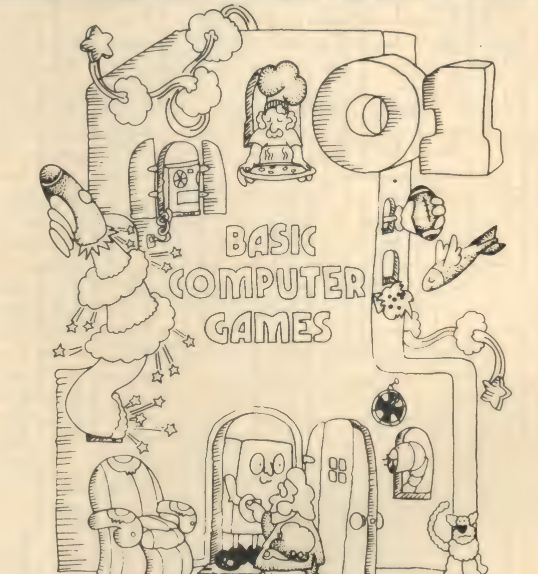
This fascinating 336-page book contains the best of the articles, fiction, foolishness, puzzles, programs, games, and reviews from Volume 2 issues of *Creative Computing* magazine. The contents are enormously diverse with something for everyone. Fifteen new computer games are described with complete listings and sample runs for each; 67 pages are devoted to puzzles, problems, programs, and things to actually do. Frederik Pohl drops in for a visit along with 10 other super storytellers. And much more! The staggering diversity of the book can really only be grasped by examining the contents, or better yet, the book itself.

Price is $8.95 plus $0.75 shipping and handling in the USA ($9.70 total); outside USA, add $1.00 ($10.70 total). Individual orders must be prepaid. Creative Computing Press, Attn: Helen P.O. Box 789-M, Morristown, NJ 07960.

THE BEST OF BYTE — VOL. 1

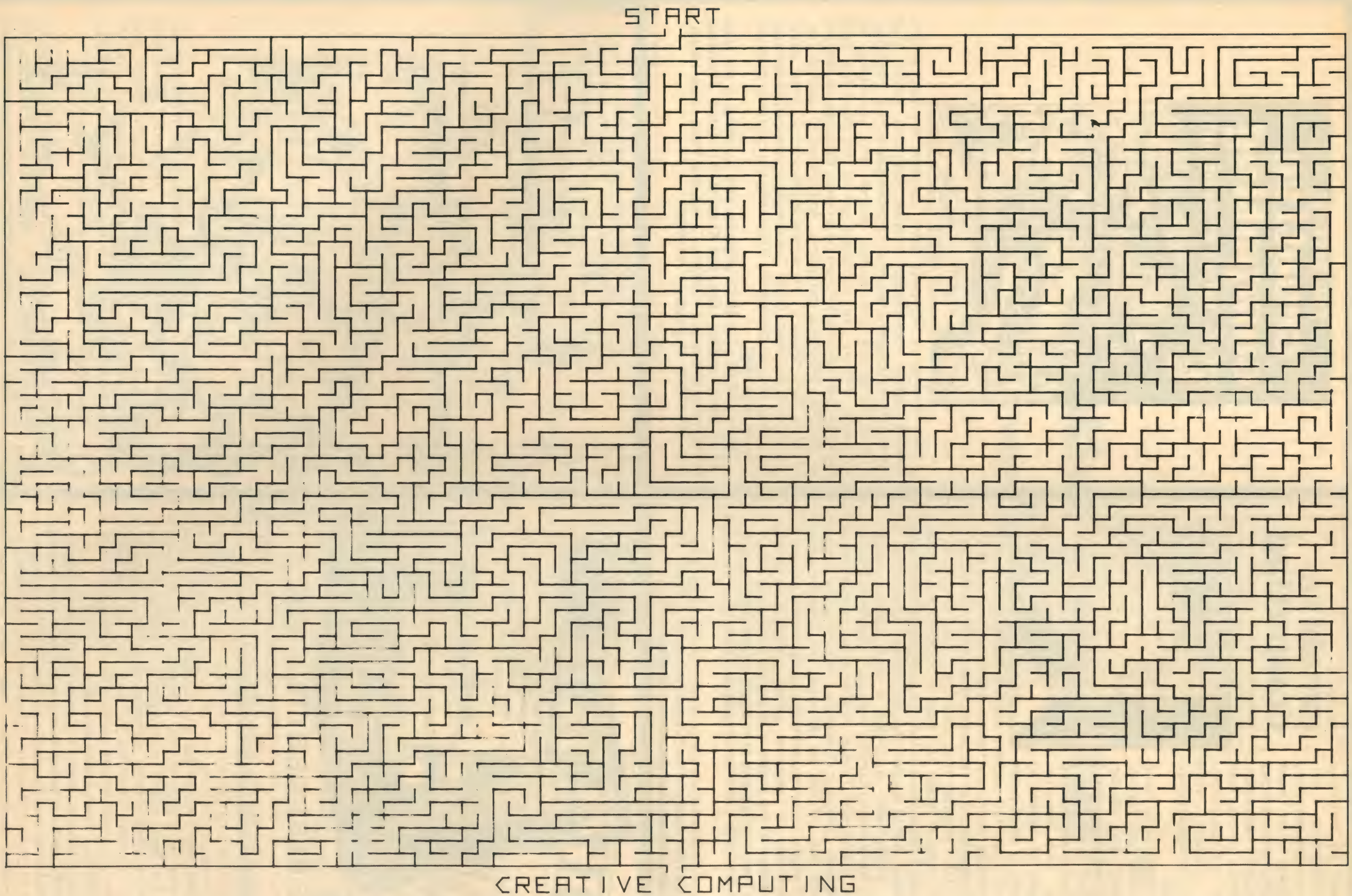
The Best of Byte - Volume 1 is a 384-page blockbuster of a book which contains the majority of material from the first 12 issues of *Byte* magazine. 146 pages are devoted to "Hardware" and are cram full of how-to articles on everything from TV displays to joysticks to cassette interfaces. The section on computer kits describes building 7 major kits. But hardware without software might as well be a boat anchor, so there are 125 pages of "Software and Applications" ranging from on-line debuggers to games to a complete small business accounting system. A section on "Theory" examines the how and why behind the circuits and programs, and a final section "Opinion" looks at where this explosive new hobby is heading.

The Best of Byte - Volume 1 is edited by Carl Helmers and David Ahl and published by Creative Computing Press. Price in the US is $11.95 plus $1.00 shipping and handling ($12.95 total); foreign orders add $1.00 ($13.95 total). Orders from individuals must be prepaid. Creative Computing Press, Attn: Helen , P.O. Box 789-M, Morristown, NJ 07960. Allow 8 weeks for delivery.



101 BASIC Computer Games is the most popular book of computer games in the world. Every program in the book has been thoroughly tested and appears with a complete listing, sample run, and descriptive write-up. All you need add is a BASIC-speaking computer and you're set to go.

101 BASIC Computer Games. Edited by David H. Ahl. 248 pages. 8½x11 paperbound. $7.50 plus 75¢ postage and handling ($8.25 total) from Creative Computing, P.O. Box 789-M, Morristown, NJ 07960.



Here is a maze that was computed and plotted by a computer. The program that generated the maze was written in BASIC. It begins by asking for the name of the maze and then for the size (it may be up to 85 by 65). Each maze that is generated will be different if given a different name and/or size.

The program was written for the HP 9830A desk top computer and the HP 9866 plotter. The program generates the maze by making a random walk through the area until it can no longer walk. It then starts walking from another point. It walks until it has visited each and every room. The paths it took are plotted to form the maze. For a program listing, send your request with a large self-addressed, stamped envelope to: Gary D. Sasaki, Hewlett Packard, 5301 Stevenscreek Blvd., Santa Clara, CA 95050, (408) 246-4300

BOOKS BOOKS

A Guided Tour of Computer Programming In Basic

Tom Dwyer and Michael Kaufman. "This is a fine book, mainly for young people, but of value for everyone, full of detail, many examples (including programs for hotel and airline reservations systems, and payroll), with much thought having been given to the use of graphics in teaching. This is the best of the introductory texts on BASIC."—*Creative Computing* Large format. 156 pp. $4.80 [8L]

Game Playing With the Computer, 2nd Ed.

Donald Spencer. Over 70 games, puzzles, and mathematical recreations for the computer. Over 25 games in BASIC and FORTRAN are included complete with descriptions, flowcharts, and output. Also includes a fascinating account of the history of game-playing machines, right up to today's computer war games. Lots of "how-to" information for applying mathematical concepts to writing your own games. 320 pp. 1976 $16.95 [8S]

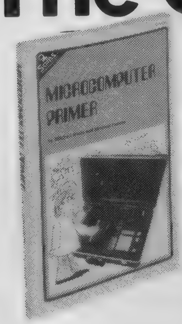
Games, Tricks and Puzzles For A Hand Calculator

Wally Judd. This book is a necessity for anyone who owns or intends to buy a hand calculator, from the most sophisticated (the HP65, for example) to the basic "four banger." 110 pp. $2.95 [8D]

Fun & Games with the Computer

Ted Sage. "This book is designed as a text for a one-semester course in computer programming using the BASIC language. The programs used as illustrations and exercises are games rather than mathematical algorithms, in order to make the book appealing and accessible to more students. The text is well written, with many excellent sample programs. Highly recommended." —*The Mathematics Teacher* 351 pp. $6.95 [8B]

8 New Books From The Creative Computing Library



Microcomputer Primer

Mitchell Waite and *Michael Pardee.* Introduces the beginner to the basic principles of microcomputers. Discusses the five main parts of a computer—the central processing unit, memory, input/output interfaces, and programs. Explains current available microcomputers. 224 pp. $7.95 [4A]

How To Buy and Use Minicomputers and Microcomputers

William Barden, Jr. Includes guides for selection, buying and programming your own system with detailed descriptions of what's currently on the market. Discusses both hardware and software, peripherals, and gives an explanation of both practical and recreational uses. 240 pp. $9.95 [4C]

TTL Cookbook

Don Lancaster. This complete, detailed guide to transistor-transistor logic (TTL) explains what it is, how it works and how to use it. Discusses practical applications such as a digital counter and display system, events counter, and electronic stop watch. 336 pp. $8.95 [9R]

Getting Involved With Your Own Computer

Les Solomon and *Stanley Veit.* Here is one of the first books on microcomputers that requires no previous knowledge of electronics or computer programming. It tells you where to find needed information, explains basic concepts, and summarizes existing systems. Good place for the neophyte to begin. 216 pp. $5.95 [9N]

Microcomputer Design

Donald P. Martin. This book is well-suited for the engineer who's designing microprocessors into his company's products. Not just block diagrams or vague theory, but dozens of practical circuits with schematics for CPUs based on 8080 and 8008 chips. Includes interfacing to A/D, D/A, LED digits, UARTs, teletypewriters. Over 440 pp. $14.95 [9P]

TV Typewriter Cookbook

Don Lancaster. An in-depth coverage of tv typewriters (tvt's)—a truly low-cost microcomputer and small-systems display interface. Completely illustrates how you can do it yourself. 256 pp. $9.95

CMOS Cookbook

Don Lancaster. Tells all you need to know to understand and profit from working with this digital logic family. It's inexpensive and genuinely fun to us. Explains CMOS—how it works and how to power it. Plus usage rules, state testing, bread boarding, interface, and other basics. Filled with practial applications. 416 pp. $9.95 [9S]

How To Program Microcomputers

William Barden, Jr. Explains assembly-language programming of microcomputers based on the Intel 8080, Motorola MC6800, and MOS Technology MCS6502 microprocessors. Basic concepts, architecture and operations discussed. Provides precanned operations. 256 pp. $8.95 [4B]

Send orders to Creative Computing, P.O. Box 789-M, Morristown, N.J. 07960. For faster service call Toll-free (800) 631-8112, or New Jersey call (201) 540-0445.

creative computing

MAGAZINE SUBSCRIPTIONS

Term	US	Foreign Surface	Foreign Air
1-Year	☐ $ 8	☐ $12	☐ $ 20
2-Year	☐ 15	☐ 23	☐ 39
3-Year	☐ 21	☐ 33	☐ 57
Lifetime	☐ 300	☐ 400	☐ 600
Vol. 1 Bound	☐ 10	☐ 12	☐ 15
Vol. 2 Bound	☐ 10	☐ 12	☐ 15

☐ New ☐ Renewal ☐ Gift

BOOKS AND MERCHANDISE

Quan	Cat	Descriptions	Price

Books shipping charge $1.00 USA, $2.00 Foreign ________

NJ Residents add 5% sales tax ________

TOTAL (magazines and books) ________

☐ Cash, check, or M.O. enclosed

☐ BankAmericard } Card No. ________

☐ Master Charge } Expiration date ________

☐ Please bill me ($1.00 billing fee will be added)
Book orders from individuals must be prepaid.

Your name and address:

Name ________

Address ________

City ________ State ________ Zip ________

Send gift to:

Name ________

Address ________

City ________ State ________ Zip ________

COMPUTER CONTROL

BY MICHAEL R. VITALE

What happens when a computer designed to accept thought patterns as input tries to output thought patterns of its own?

999 END
READY
RUN
*** PLEASE STAND BY ***

The cursor blinked white against the green screen of the cathode ray tube. "Oh, no," the student thought to herself, "what now?" She glanced up over the terminal towards the glass wall of the computer room, attempting to see within any reason for the halt in service. She scratched the place behind her left ear where the electrode rubbed against her skin and adjusted the metal cap slightly. Her eyes returned to the cursor, flicked away, then returned and remained fixed. A few seconds later she rose, took the cap from atop her head, and set it down on the chair. She ran around the terminal once, flopped to the floor, and did ten pushups. Then she ran around the terminal twice, flopped to the floor, and did ten more pushups. Then she ran around the terminal three times, puffing slightly now, flopped again to the floor, and did another ten pushups.

After circling the terminal 55 times and doing a total of 100 pushups, the girl picked the cap off the chair, replaced it on her head, and slumped wearily behind the screen.

CONTINUE

appeared before the girl's glazed eyes. From somewhere behind the glass wall arose a deep chuckle.

* * * * *

The intercom buzzed twice. The president took his feet off the desk, set down the paperback book, and flicked the switch.

"Professor Eckert to see you, sir."

The president mentally ran through the names of the school's 253 faculty — *his* 253, as he liked to think of them — but could not place Eckert. "Professor Eckert?"

"The head of the computer center, sir."

"Oh, Eckert, What the hell does *he* want? If it's about the budget tell him I'm sorry but not another dime for his damn tape packs or whatever."

"He says it's an emergency, sir."

"I'll bet it's an emergency. You tell that twerp that if his damn machine has screwed up the payroll again I'll send him straight back to typewriter repair school."

"He seems quite excited, sir."

"He's probably finally figured out how to get that toy of his to do some useful work for a change. Give me two minutes." The president pushed aside several overflowing manila folders, some old issues of the campus newspaper, a half-eaten chocolate bar, and a report from a dean who had left the college seven months before. He picked the top binder from a large pile and opened to a page at random. He flicked the intercom switch again. "Send Eckert in."

The door opened and a small, nervous man with wildly tousled hair took a few steps into the cavernous office.

"Well, Eckert, good to see you! I've just been reading over your latest outprint."

"Printout, sir."

"Well, whatever. I see that the number of students has increased 2.84736 percent over the past three years, while the number of faculty has decreased by 7.5846 percent and the number of administrators has increased by 14.596 percent."

"That's true, sir. Actually I came to discuss something else."

"I hope it's not the budget again, Eckert. I *am* sorry about cutting you back this year, but I thought that with that large grant from the National Science Foundation you could get by without so much help from the college."

"It's the NSF project that I wanted to talk about, sir."

"You've made progress, I hope? It's almost time for our annual report to the government."

"Yes, we've made progress." Eckert sighed and gazed at the carpet.

"Well, what is it, then?"

"As you recall, sir, the grant was to support an experimental project in artificial intelligence."

"Of course I remember." He didn't. "I thought at the time that developing some artificial intelligence among the faculty was a good idea, since they seem to be somewhat lacking in the natural kind."

Eckert forced a smile. He had heard the joke before. "We were trying to work out a means by which instructions could be entered directly into a computer without the necessity of punching cards of typing on a terminal. It's a programmer's dream!" Eckert's glowing eyes moved briefly upward, but were forced back to the carpet by the president's uncomprehending stare. "During the early seventies, various means of entering information by voice were developed, but . . ." The phone rang.

"Excuse me a moment, Eckert." The president picked up the receiver. "Yes?" The president listened for a moment. "Well, you tell that bastard that he damn well *better* get those lawns cut, and by this afternoon, or I'll fire his ass." The president listened again. "Well, I suppose you're right. Talk to the union steward, then. Ask him to be reasonable, for once." The president slammed the phone back into the cradle. "Dammit, nobody ever does *anything* around this place. Now where were we?"

"I was explaining the background of the NSF project."

"Oh, right. Go ahead."

"The vocal techniques worked fairly well, but they were very sensitive to slight changes in tone — a strange voice threw them off completely, and even a sore throat could cause trouble. Besides, they were *slow*. A human can think much faster than he can speak."

"Seems to me we've got quite a few people around here of whom exactly the opposite is true."

Eckert forced another smile. He had heard that one before, too. "In any case, we wanted to develop a technique for entering information directly from the brain into the computer. No typing, not even any speaking. So we formed an artificial intelligence lab, applied for a government grant, and began working. That was three and a half years ago."

"Yes, you took over the old bus garage, didn't you?"

"The old boiler room, sir, when the college installed the solar panels. But we've gotten most of the soot out by now. You'd hardly recognize the place. Anyway, eventually we developed a sort of metal beanie through which the brain could transmit information directly to the computer. The data channel was designed to be one-way — the computer could send information back to a fast terminal device, as usual." Eckert paused, sighed, and resumed speaking. "We gathered some student volunteers, all experienced programmers, and turned them loose with the beanies. Until last

week everything was going fine."

"Yes?" The president knew better then to ask directly what had gone wrong. He passed his hand over his forehead, trying to wipe away visions of dazzled students with hair standing on end and eyeballs rolling idly.

"Then we began noticing that the participants were behaving strangely. First they began volunteering to sweep out the computer room."

"Sounds good to me."

"Yes, but not every hour. Then they began offering to oil the tape drives, vacuum the disk packs, and empty the chad box on the card punch. Finally we had to lock the machine room door so we could use the equipment."

"What are you suggesting, Eckert?" The president's tone had changed.

"Well, apparently the computer has grown . . ."

"GROWN?"

"Well, developed, then, another circuit, which somehow lets it program the students." Eckert stopped.

The president tried to absorb the news. "So we load the thing with calculus, plug in all the students, and cut back the math department? Well done, Eckert!"

"No, sir, I'm afraid that's not exactly what I wanted to say."

"What's the problem, man?"

"You see, we don't know how to control *what* the students are being programmed to do. The computer seems to be doing it on its own."

"Son of a bitch." The president savored each syllable. Having the payroll screwed up was nothing.

"I wanted to alert you to the, uh, situation as soon as I was sure that it was really happening. I have my best programmer working on it right now," Eckert lied.

"Your best programmer? Why can't you do it yourself?"

"Well, I'd like to, but I have the daily, bi-weekly, weekly, semi-monthly, monthly, quarterly, and annual reports to get out, besides the new grading system and the classroom space evaluation."

"Of course. Well, all I can say is you'd better get some action . . ." There was a scuffling noise in the outer office, the sound of a body being pushed against a wall, then the door was jerked open. A large, bearded student ran across the room, pushed a custard pie into the president's face, and left as quickly as he had come.

* * * * *

Eckert pulled open the door of the computer center, pushed past a student doing frantic jumping jacks, and edged between two girls playing a game of imaginary tennis. He entered his secretary's office. "Has Tom arrived yet?"

"He's waiting for you inside."

Eckert tried to open his door. The knob came off in his hand. He sighed. "The door is still broken."

"I know. I've asked to have it fixed, but there's no . . ."

No money in the budget, Eckert thought, and rapped on the door. It was opened from the inside by an emaciated, acne-scarred student wearing pants several inches too short. He was rubbing sleep from his eyes with his free hand. "Hello, Tom. Thanks for coming over. I know you were up late last night straightening out that problem with the Fortran compiler."

"I found a bug at 04756 octal — a zop fault which under certain circumstances prevented the J-block from linking up with the permfile."

"Well, thanks for fixing it." Eckert carefully closed the door behind him. "Tom, I wanted to talk to you about the artificial intelligence project."

"You haven't let me work on that one yet."

"I know, Tom. I thought we could use you more effectively somewhere else." The true was, Eckert mused, that Tom seemed to relate a lot better to machines than to anything which involved people. "But we need your help now." Eckert recounted the problem in detail, emphasizing the machine-language programming techniques which had been used. "We've got a list of all the students who participated in the project. Of course we'll have to scrap the whole thing now, but first we have to find some way of, well, de-programming them. That's where you come in. Do you think you can handle it?"

"I'll try." This was Tom's way of showing enthusiasm. "Can you give me access to the documentation?"

"Yes. My secretary has all the information on her desk."

"OK. I'll try to find the problem. Then we can round everybody up and unload them all at once. I'll be in touch." Tom reached for the door, ignored Eckert's warning cry, and pulled off the other doorknob. He slammed the door, locking Eckert into the office, took a large pile of jumbled papers off the secretary's desk, and left the center.

* * * * *

Eckert stood nervously behind the control console. "Are you sure this is going to work?" Tom did not bother to look up from the keyboard.

"We're ready to go."

Eckert left the machine room and surveyed the small group of waiting students. "All right, everyone, we're going to try a new type of experiment." Eckert felt a momentary twinge of guilt for not having told the students what had happened — or what he hoped was going to happen now. "Please put on your beanies and sit down behind a terminal." The students scuffled noisily into their places, and Eckert walked behind the row of terminals to be sure that everything was ready. He signalled to Tom through the glass. Tom typed a few characters and looked up expectantly.

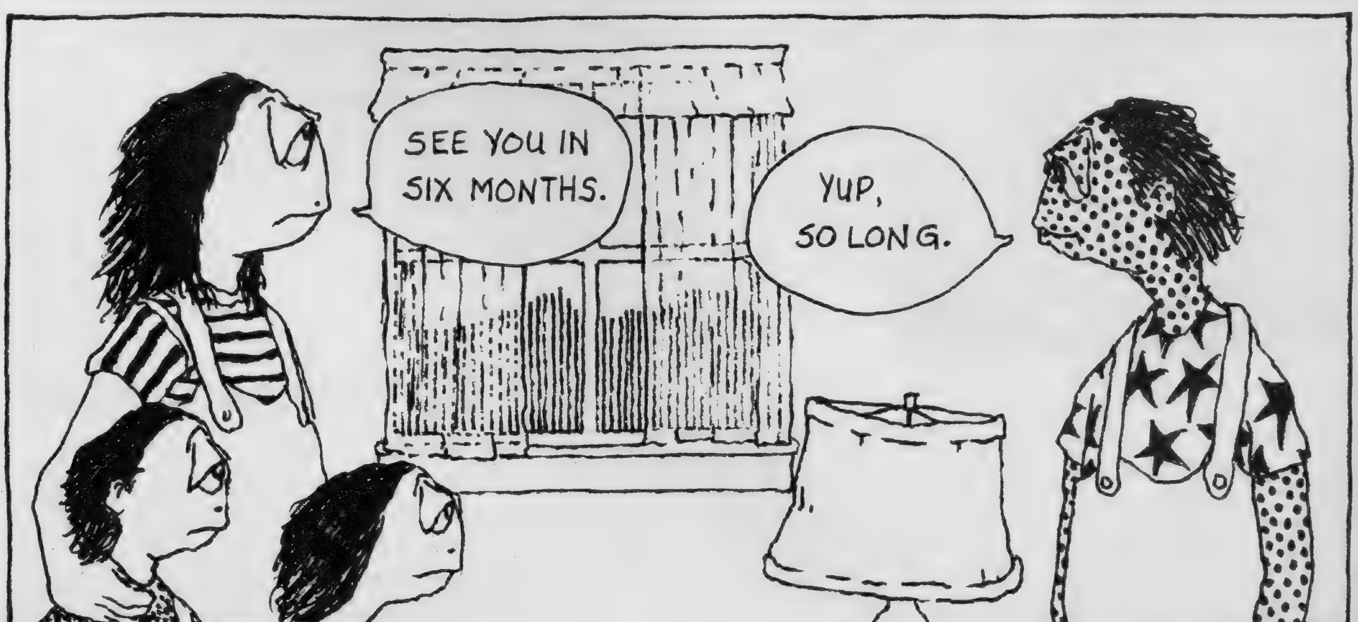
* * * * *

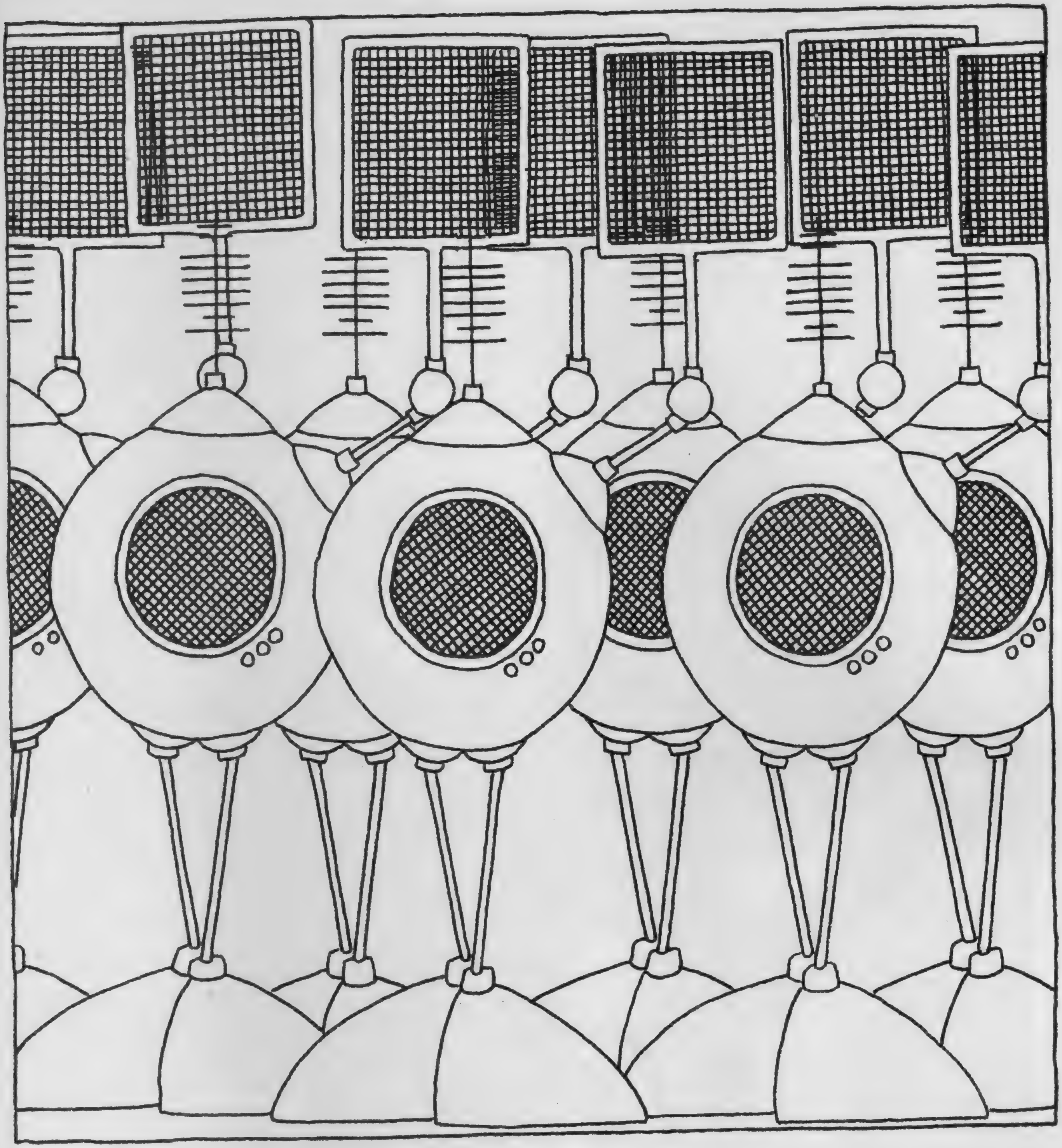
There was a knock at the dormitory-room door. Tom looked up from the latest issue of *Computing Reviews*, glanced at his watch, and said, "Come in, Cindy." The door opened, and a lithe, tanned girl with long brown hair and a perplexed expression entered. Cindy removed her pullover, kicked off her shoes, and slid out of her bluejeans. Tom got up from the desk, locked the door. and switched off the light. "I've been expecting you." ■

NORDS

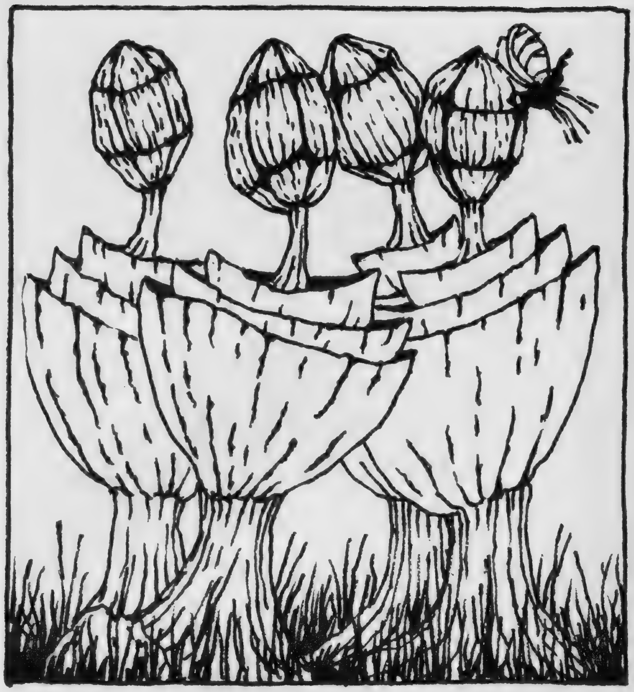
The Nords walked around with their eyes on the sky and their hands on large swatters. There was a good reason for this: the skies were full of large bugs called snits.

The snits were not considered a blessing. The snits had a bite that felt like the caress of a chain saw. A Nord bitten by a snit broke out in spots and developed an odor so horrendous that it was called The Curse. The condition lasted six months. It was six months in isolation for no one could stand the odor.

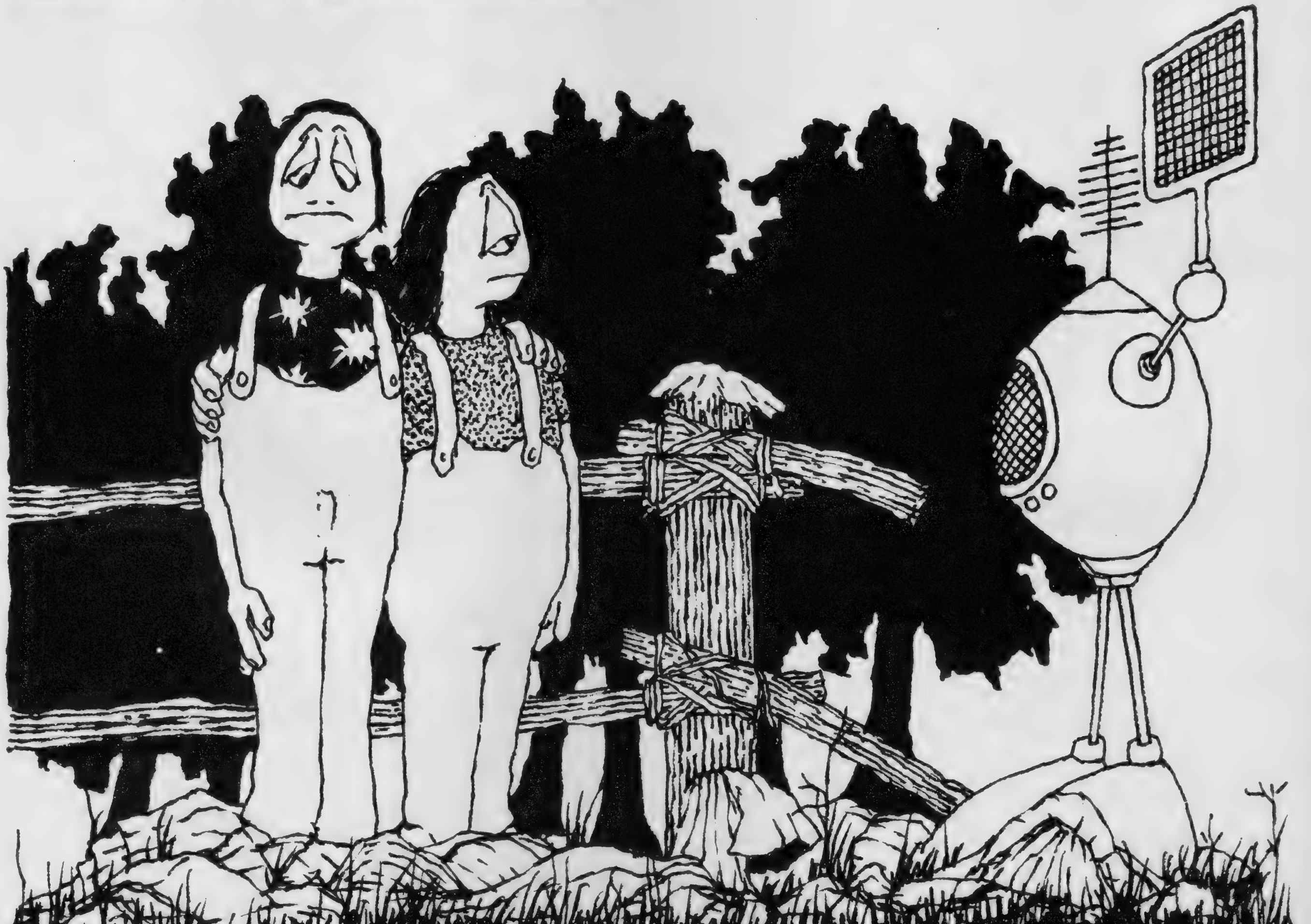




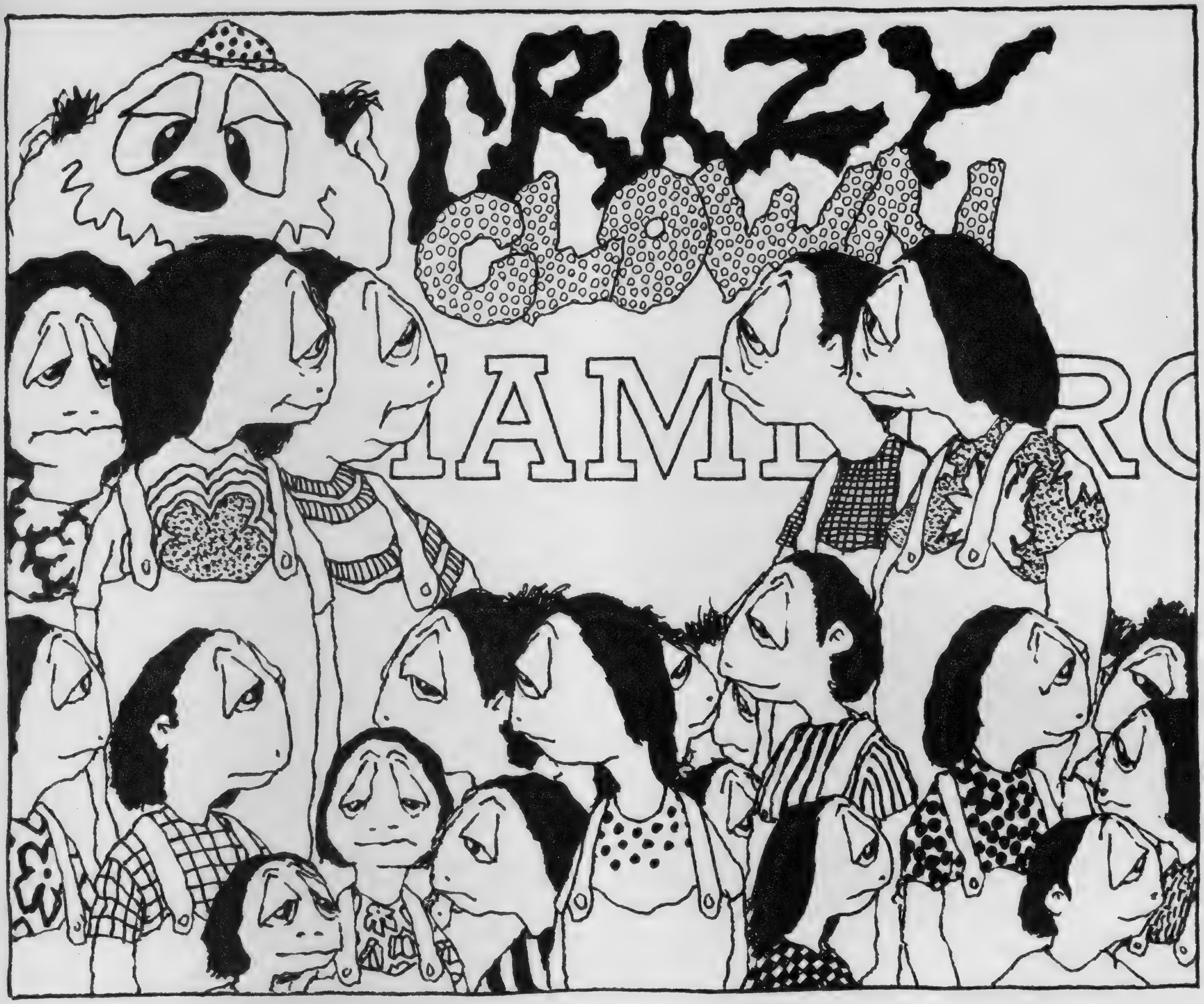
Then the Nords developed the technology to rid themselves of the snits. It was in the form of a robot equipped with a large swatter. The Nords built an army of them and programmed them to kill all snits. The Nords figured heaven was just around the corner.



The robots began by chasing snits all over creation but it wasn't long before the robots discovered that the snits had only one source of food: the leaves of the Maunt plants. Programmed to destroy all snits and computing cause and effect as being one and the same, the robots used their swatters to reduce all the Maunts to mangled pulp.

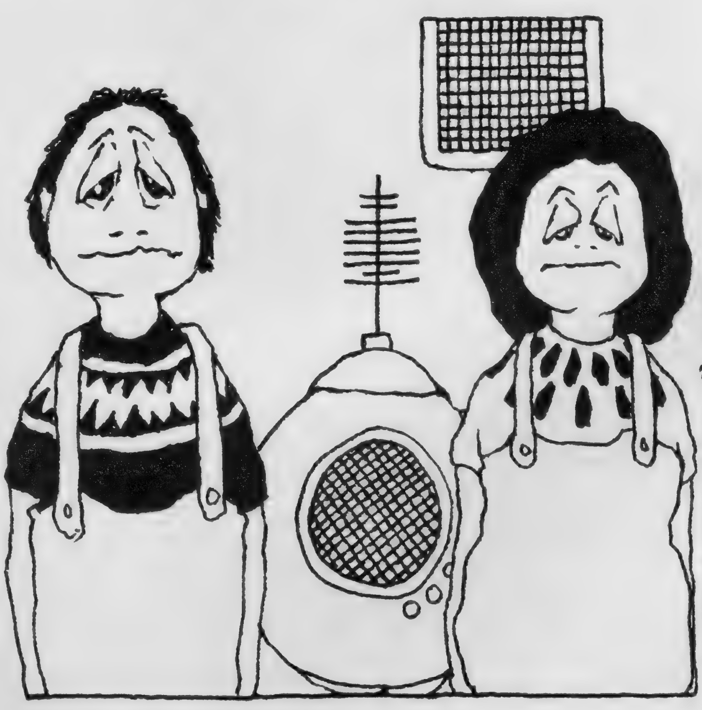


That was unfortunate. What the robots didn't know (and couldn't have known since the information wasn't included in their programs) was that the seeds of the Maunts were the only source of birth control the Nords had.



The population increase was immediate and immense.

Desperation set in. Laws were passed. Mating was restricted to three days a year. The robots were reprogrammed and used as police. One was assigned to each home. Any couple attempting to mate on any other than the three designated days faced the possibility of being reduced to mangled pulp.



reviews...

The Dragons of Eden: Speculations on the Evolution of Human Intelligence. Carl Sagan. New York: Random House. 263 pp. Hardback. $8.95. 1977.

How You Can Learn to Live with Computers. Harry Kleinberg. New York: Lippincott. 216 pp. Hardback. $8.95. 1977.

Getting Involved with Your Own Computer: A Guide for Beginners. Leslie Solomon and Stanley Veit. 216 pp. Hardback. $9.95; paperback, $5.95. 1977. From: Ridley Enslow Publishers, 60 Crescent Place, Box 301, Short Hills, NJ 07078 or *Creative Computing* book service.

Since it's all the current rage, and is supposedly science fiction, and *is* available in book form, I feel somewhat compelled to discuss *Star Wars.* I guess I can justify this by talking about the use of computer technology in the movie.

Star Wars is a well-done, fast-paced adventure flick with something to enthrall everyone. The visual effects are stunning and superbly done, the plot won't confuse you (synopsis: good guys, wearing white, versus bad guys, wearing black), you'll have no trouble deciding which side to cheer for (hint: the damsel in distress is wearing white), the violence is clean, and the sex is limited to good-luck kisses. The scale comes across as being properly huge. I kept expecting the minions of Boskone and a Gray Lensman or two to pop up at any moment.

However, in spite of the fact that you become very involved in the movie while watching it, *Star Wars* falls kinda flat when you think about it afterward. It is decidedly not *2001* and not close even to *Star Trek*. None of that almost mystical quality of gaining new insights which made *2001* such a cherished experience, and none of the believability which makes *Star Trek* such a desirable future in which to daydream. *Star Wars* is about as profound, believable, and desirable as a James Bond movie.

Anyway, I said I was going to talk about the technology. Actually I'm going to say something about the computer technology *not* used. The universe presented in *Star Wars* is nothing short of scatter-brained, which makes for very poor science fiction. When looked at closely, the movie disintegrates into an oddball collection of incompletely thought-out ideas held together with that wonder glue, action. Why do the Jawas run around Tatooine in a motorized septic tank and why is the surface of the planet apparently littered with cast-off or wrecked 'droids? How can the Millenium Falcon take off from a planetary surface? Why oh why does the Death Star have a trash masher in which a tentacled monster lives? Arghh! Maybe there are reasons, but they sure weren't given.

reviews...

And from our point of view, the biggest question must be why the techology required for C-3PO is nowhere else in evidence. Huh? C-3P0 has very advanced voice recognition and synthesis, an acute artificial intelligence, and is obviously the produce of ultramicrominiaturization. So why are they still punching buttons to control starships, why do people man turret-mounted lasers, fly fighters which are moving far too quickly for human reaction time to control (a lot happens in 120 milliseconds at those speeds, my friends), and ... Arghh! Of course one can always hope that there is a ban on using advanced cybernetic technology in warfare, but probably they just didn't think of it. The book is even worse than the movie in its inept use of technology. I hope they listen to some competent technical advice for the sequels.

Unfortunately, I just can't stand to boycott the movie for its stupidity. What else is there? At least it's better than, ick, *Space: 1999.* Sigh. May the Farce be with you.

Let's talk about another best seller now, and a somewhat unexpected one, *The Dragons of Eden.* Carl Sagan is, as he readily admits, going a little far afield from his usual realm of astronomy and extraterrestrial intelligence to write a book dealing primarily with the evolution of the human brain, but he does it very well, producing a quite readable and quite fascinating book. He gives a good overview, in reasonably non-technical terms and with several helpful charts and illustrations, of what is known for certain about human evolution, but the real fascination is in the speculations hinted at by the subtitle. Abundant food for thought is presented, especially in the chapter, "The Future Evolution of the Brain."

What makes it so interesting that this book has gained best-seller status is that Sagan adopts, and not simply as metaphor, the model of the brain as an information processing system and nothing more. He states boldly in his introduction that "both because of the clear trend in the recent history of biology and because there is not one shred of evidence to support it, I will not in these pages entertain any hypothesis on what used to be called the mind-body dualism, the idea that inhabiting the matter of the body is something made of quite different stuff, called mind." Computer terminology such as "hard-wired" and "programmed" is used extensively throughout the book in discussing the functioning of human beings. This is a pretty heavy idea to lay on The General Public, but I haven't yet heard of any bible-belt burnings of the book.

The Dragons of Eden is just chock-full of ideas and statements and plausible assertions which give one pause to think. It is an excellent catalyst to discussion. I heartily recommend this book to the curious, for any course touching on Artificial Intelligence or human intelligence, and I especially recommend it to those of you who while away cold, lonely nights trying to answer that question of questions: "What am I?" If we keep working on it, one of these days we're going to answer that question.

Now let's consider another in what is bound to be an unending stream of books on the general topic of Explaining Computers to The Masses. Harry Kleinberg has written an interesting book called *How You Can Learn to Live with Computers.* It is one of the clearest explanations of what computers are and what they can be used for, that I have yet read. It may possibly rank with *The Peter Principle* in pointing out an obvious fact which has gone unnoticed for lack of being formalized.

After devoting the first half of the book to a simple explanation of what is, afterall, a simple device, Kleinberg goes into a rather roundabout discussion of what the computer is, with the conclusion that the computer is a logic device. Hardly a startling conclusion, right? He then points out that, if a task requires more than logic, present computers can't do it. Simple? Sure. Building on that unarguable fact, Kleinberg introduces a concept he calls the K Vector, which is a way of convincing you

that, assuming you're worried about such things, you cannot be completely replaced by a computer and can probably estimate how much of your job function *can* be replaced by a computer. (*I* feel secure — a reviewer can't possibly be replaced by a computer. What's logical about reviewing?)

Kleinberg's book is slightly cautious, very reassuring, and quite helpful in gaining an understanding of computers on a gut level. This is a good book to give to a person still suffering from the traditional misconceptions about computers. In fact, a lot of people in the AI community might profit from reading this book. It makes it very clear that present computers simply cannot ever be intelligent. Sorry. Just isn't possible. There is more to intelligence than logic.

I kept saying "present" computers for a reason. Thirty years is not long in the evolution of a machine intelligence. Don't give up hope. The time of the machines will come.

The new book by Leslie Solomon *(Popular Electronics),* and Stanley Veit (Computer Mart of New York), *Getting Involved with Your Own Computer,* is for people who haven't gone much farther than wondering what personal computing is all about. An introduction is given to further sources of information, basic vocabulary and concepts are briefly explained, and most of the popular personal computing systems now on the market are summarized. The explanations of programming, software, and "What Can You Do with a Computer?" are rather muddled, but that is something from which the entire field of personal computing is presently suffering. The glossary is one of the worst I've seen. Still, this book is not a bad place to start.

Also out and floating around is the *Interim Report on Humane Computing: A Survey of its Theory and Practice.* This is a project trying to "identify the important issues in the use of computers for the enhancement of our lives with a view to encouraging positive trends in this direction." The interim report is mainly a reaction to, and compilation of, responses to a request for ideas and information which Clement ran in most of the personal computing magazines last year. The present report is still in the food-for-thought stage. If you're interested in this area, Andrew Clement can be reached at 789 West 18th Avenue, Vancouver B.C., Canada V5Z 1W1.

You'll have noticed that I've stopped asking for volunteer reviewers. Right now I'm swamped, with around 75 willing people in my file and obviously not that many books to send out. I'll let you know when I need people again.

Magic Squares. Paul Calter. New York: Thomas Nelson Inc. 143 pp., hardbound. $7.95. 1977.

Magic Squares, by Paul Calter, is ... not your normal book. On the one hand, it's a rather tongue-in-cheeky detective story featuring Mat Inverse, CAD (Computer Assisted Detective), whose sets out in a somewhat bumbling manner to foil a plot to force the world to renounce its use of computers and return to the time-tested ways of the Pythagoreans, and on the other hand it is a book full of puzzles and numerology. Solving the puzzles, most of which are programming problems, is not necessary to follow the plot. In spite of Calter's tendency toward atrocious puns, he has written an entertaining little story.

Compilation of State & Federal Privacy Laws. Robert Smith and Keith Snyder. *Privacy Journal,* P.O. Box 8844, Washington, D.C. 20003. 8½ x 22 paperback, 215 pp. $12.50. 1977.

The only such conpendium I know of on privacy and confidentiality statutes and legislation.

My Friend — The Computer. Jean Rice. T.S. Denison & Company, Inc., 5100 West 82nd. St., Minneapolis, Minn. 55437. 85 pp. (8½ x 11, paper). $2.95. 1976.

My Friend — The Computer introduces the young reader to computers and their uses. The book is divided into seven parts covering the following topics: Part one discusses what a computer is and the notion of time-sharing. Part two discusses some common uses of computers in medicine, industry, etc.. Part three presents a brief history of computers. The fourth part outlines the various parts or units of a computer. Part five deals with input, output, and memory devices. In the sixth part the notion of organizing thoughts via flowcharts is presented. The last part of the book deals with programs and the BASIC language. (This is not meant to be a BASIC primer, but rather an illustration of a programming language thru a few simple examples.) The book contains two appendices, one dealing with Teletype operation and the other with preparation of paper tapes. The text concludes with a glossary of simple definitions of computer terms. Following each section is a set of simple questions (and answers) which would indicate understanding of the material presented.

A strong point of the book is the lists of student activities associated with each section. The book is exceptionally well written, and uses cartoons and illustrations very effectively.

This book can be highly recommended for use in a unit at the elementary school level (grades 5 - 7). No elementary school library should be without it.

Bruce W. DeYoung
Oakland, N.J.

There is now an excellent Teacher's Guide to accompany Jean Rice's book, containing text, spirit masters and overhead transparencies — JL

Systems and Programming Standards. Susan Wooldridge. Mason Charter. 189 pp., Hardbound. $12.50. 1977.

For anyone who is new at being a DP manager, this is almost a must topic; and this new text covers the subject matter with clear, concise examples and careful planning using a step-by-step approach. In addition to management, Ms. Wooldridge feels it is to everyone's advantage that all DP personnel know something about standards design.

Ms. Wooldridge's style is flowing and easily read even for a topic as cut-and-dried as standards. She is obviously knowledgeable about the subject of DP management, and convincingly argues for the need for good, up-to-date standards, supporting this as basically a management function.

As Ms. Wooldridge herself warns, this book is definitely geared toward the professional "in commerce or government... heir to the practical difficulties of daily reality" in that environment (the student is cautioned to make necessary adjustments when using this text for courses). So for those responsible for standards, this text could help. But as a must reference text for the shelf of every programmer, analyst, or even DP manager, I would not particularly recommend it.

Blaise W. Liffick
Rochester, NY

Some Common BASIC Programs. Lon Poole & Mary Borchers. Adam Osborne & Associates, Inc., Berkeley, CA. 192 pp. $7.50. 1977.

This book contains 76 programs that "perform a variety of common, practical tasks. "Most of the programs are concerned with business, mathematics, or statistics, but there are a few unusual ones.

Each program starts with a short description, followed by a run and a listing. Also included are optional additions that may be used to change the program's function a bit.

The style of the writing is ordinary and concise. The programming techniques are standard ones that can be adapted to most versions of BASIC. The programs can be used by someone who has no knowledge of BASIC.

This would be a useful book for people who are not interested in programming, but need a computer to perform routine tasks, such as plotting equations or finding standard deviations. Most *Creative Computing* readers would probably rather write their own programs, as most of them are fairly simple. These are the type of programs that everybody writes when they are learning BASIC. Still, it is a useful reference book, especially for people who don't have the time or knowledge enough to write their own programs.

David Rothenberg
Westport, CT

Interactive Data Analysis. Donald R. McNeil. John Wiley & Sons, Inc., New York. 186 pp, paperbound. $9.95. 1977.

Interactive Data Analysis is a statistics textbook with a twist. The typical college-level approach starts with elementary probability theory and develops the concepts of statistical inference. McNeil instead starts with what he calls exploratory data analysis — analyzing data to find its inherent structure, to facilitate later model building.

The reason for this approach, explains McNeil, is that "in scientific work, one almost always begins with the data, and when structures in the data are found to be statistically significant, a model or theory is sought to explain those patterns."

Among McNeil's somewhat unorthodox, yet quite useful, analytical tools are new kinds of data displays, such as the "stem-and-leaf plot," which looks somewhat like a histogram but contains much more information about the sample, and the "box plot," which provides a very compact graphical representation of a sample's medians and quartiles (or other desired partitions).

The author has provided programs in both FORTRAN and APL versions to produce the displays and all the data transformations discussed in the book. (Although he has emphasized the use of the computer, all the tiques and exercises can be done without much difficulty using pencil and paper.) And although he assumes no prior knowledge of statistics (or really of computing), his novel approach of "numerical detective work" will also be interesting to students who have learned more traditional statistics.

Thomas A. Gutnick
Arlington, VA

FORTRAN Techniques. A. Colin Day, Cambridge University Press, New York. 96 pp. $3.95, 1972.

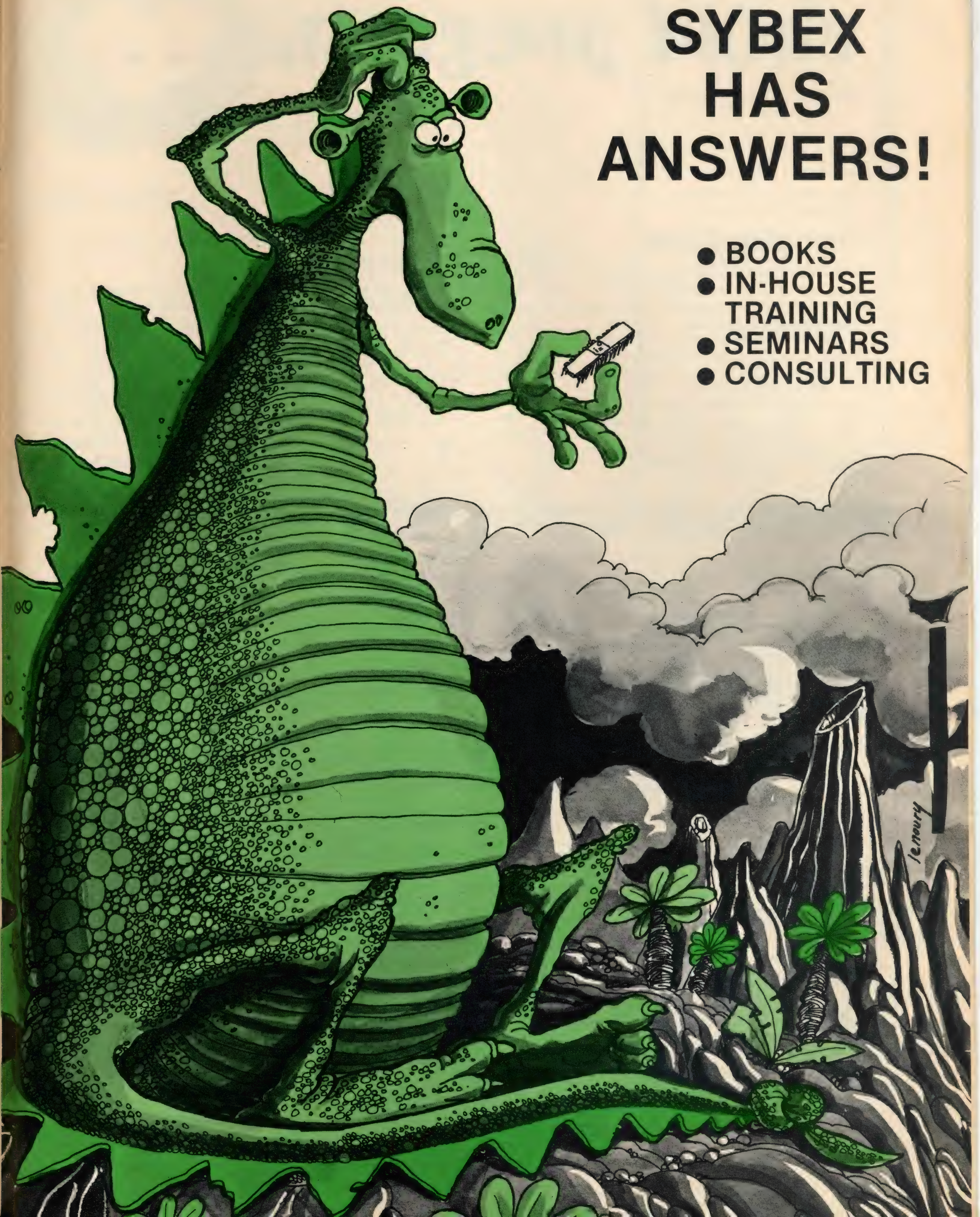
This book's subtitle is "special reference to non-numerical applications" for good reason. In a slim 96 pages, the author covers many salient points on lineprinter graphs, table searching, keyword-in-context identification, stacks and queues, list processing, and sorting. The book fills in knowledge on a set of random techniques used by FORTRAN programmers in each of these areas; it is a practical book rather than a theoretical one.

The first chapter contains descriptions of basic techniques such as flags and switches, packing and unpacking numbers, table translation, buffers, open-coded subroutines, and simple character manipulation hints which will work on even the most restrictive FORTRAN compliers. The chapter on line-printer plotting includes remarks on point, line, density, and histogram plots, the chapter on sorting includes four sorts, and the chapter on stacks and queues includes a description of simulating recursion in a FORTRAN program. The only fault I can find with the book is that Day wastes a chapter on "symbol state tables" (useful to check that data conforms to a set of syntactic rules), giving it too abstract a coverage for such a book. All in all, Day manages to pack copious hints and notes (including *examples* and diagrams) quite well into a slim but concise reference work.

Brian N. Hess
Western Springs, IL

SYBEX HAS ANSWERS!

- BOOKS
- IN-HOUSE TRAINING
- SEMINARS
- CONSULTING



short programs.....

Program Listing

Changing Bases

Here's a clever little program (which could be used as a subroutine in a larger program) to convert from one base to another, in particular from base 10 to a new base between 2 and 16. The crucial parts of the routine are found in statements 150-170 which determine how many digits are in the new number by raising the new base to successively higher powers and checking if the resultant number is larger than that to be converted. For example, if we wanted to convert 15 to base 2, statements 150-170 would change variables as follows to finally meet the condition of B^P greater than N(1).

B	P	B^P	N(1)
2	0	1	15
2	1	2	15
2	2	4	15
2	3	8	15
2	4	16	15

Hence, the routine has determined that there are 4 digits (P) in the new number.

Statements 180-210 then do the actual conversion and deposit the new number in D(1) through D(P). We're not going to tell you how it works but leave that up to you to figure out.

Thanks to Jim West of Teletype Corp. for leaving this little gem in my GE timesharing account. — DHA.

```
090 REM   BY JIM WEST, WILMETTE, IL, (312)256-1621, JAN 1977
095 DIM D(20), N(20)
096 FØR J=1 TØ 6\ READ D$(J)\ NEXT J
097 DATA "A","B","C","D","E","F"
100 PRINT "THIS PRØGRAM WILL TAKE A PØSITIVE INTEGER (BASE 10)"
110 PRINT "AND CØNVERT IT TØ ANY NEW BASE BETWEEN 2 AND 16"\PRINT
115 REM      INPUT
120 PRINT "NEW BASE =";\INPUT B
125 IF B < 2 THEN120
126 IF B > 16 THEN 120
127 IF B - INT (B) > 0 THEN 120
130 PRINT "INTEGER (BASE 10) =";\INPUT N(1)
131 IF N(1)-INT(N(1)) > 0 THEN 130
133 IF N(1) < 0 THEN 130
135 REM      CALCULATIØN
140 LET P = 0
150 IF B^P > N(1) THEN 180
160 LET P = P + 1
170 GØTØ 150
180 FØR I = 1 TØ P
190 D(I) = INT (N(I)/B^(P-I))
200 LET N(I+1) = N(I) - (D(I)*B^(P-I))
210 NEXT I
220 REM       ØUTPUT
230 PRINT\PRINT
235 PRINT N(1); "(BASE 10) =";
240 FØR I = 1 TØ P
245 IF D(I) > 9 THEN 255
250 PRINT D(I);\GØTØ 260
255 PRINT D$(D(I)-9);
260 NEXT I
265 PRINT " (BASE"; B; ")"
270 PRINT\PRINT\PRINT
300 GØTØ 120
999 END
```

The input section checks for invalid input, decimals, negative numbers and the like

The digits of the new number are put in D(1) - D(P).

Sample Run

```
THIS PRØGRAM WILL TAKE A PØSITIVE INTEGER (BASE 10)
AND CØNVERT IT TØ ANY NEW BASE BETWEEN 2 AND 16

NEW BASE =? 2
INTEGER (BASE 10) =? 1977

 1977 (BASE 10) = 1  1  1  1  0  1  1  1  0  0  1  (BASE 2 )


NEW BASE =? 8
INTEGER (BASE 10) =? 1977

 1977 (BASE 10) = 3  6  7  1  (BASE 8 )


NEW BASE =? 12
INTEGER (BASE 10) =? 1977

 1977 (BASE 10) = 1  1  8  9  (BASE 12 )


NEW BASE =? 16
INTEGER (BASE 10) =? 1977

 1977 (BASE 10) = 7  B  9  (BASE 16 )
```

FRIEND

There is a computer I would like to call friend.
Then all of my troubles and worries would end.
It is the lottery computer run by our state.
Millionaires are made by one stroke of fate.

If I were to befriend this electronic brain
My tastes again would run to champagne.
And the machine I would not soon forget,
For the two of us together make a fine duet.

I would make it some flowers made up of wire,
Feed it a voltage just a little bit higher.
Install light bulbs a tiny bit brighter,
Ease the tape tension a little bit slighter.

We millionaires are a bit touched in the head.
For my honorable intentions word now I spread,
Look soon for your newspaper story to carry
Headlines of "Computer and man soon to marry."

Edward Stewart

THE LAST BUG

'But you're out of your mind,'
They said with a shrug.
'The customer's happy—
What's one little bug'

But he was determined.
The others went home.
He spread out the program,
Deserted, alone.

The cleaning men came. The
Whole room was cluttered
With memory dumps, punch cards,
'I'm close,' he muttered.

The mumbling got louder,
'Simple deductions,
I've got it, it's right, just
Change one instruction'

It still wasn't perfect
As year followed year
And strangers would comment,
'Is that guy still here'

He died at the console
Of hunger and thirst,
Next day he was buried
Face down, nine edge first.

And the last bug in sight,
An ant passing by,
Saluted his tombstone
And whispered, 'Nice try!'

Author Unknown

(Submitted by J. Prusis,
Dearborn, Michigan)

short programs...

Delving Into Depreciation

This simple little program computes annual depreciation on a capital item using 3 different types of depreciation. The IRS generally allows any type of depreciation as long as it is consistent; in other words you can't start depreciating an item using the double declining balance method and then switch to straight line.

Why different methods of depreciation? Because depreciation serves different purposes. Say you're a car leasing firm; then most of the depreciation occurs early in the life of your capital goods (automobiles) and you might want to use the double declining balance or sum of digits method to reflect this. On the other hand, an auto service shop might buy a hoist with an expected 5-year life. Because you expect the income of the shop to increase over time you'd like to push as much of your expenses as possible to future years to offset the increased taxes. Hence, you would opt for straight-line depreciation.

To delve further into the mysteries of depreciation as an indirect source of capital and also the tax effects, I recommend almost any business finance text such as Hunt, Williams, and Donaldson: *Basic Business Finance* published by Irwin. — DHA

```
120 PRINT "ØRIGINAL CØST ØF ITEM (DØLLARS, NØ CENTS)";
130 INPUT C
140 PRINT "LIFE ØF ITEM (YEARS, NØ MØNTHS)";
150 INPUT L
160 PRINT "SCRAP VALUE (DØLLARS, NØ CENTS)";
170 INPUT S
180 PRINT
190 PRINT "YEAR","STRAIGHT","SUM ØF"," DØUBLE"
200 PRINT " ","  LINE","DIGITS","DECLINING"
210 PRINT
220 V=C-S
230 D1=V/L
240 Y=((L+1)/2)*L
250 Z=L
260 FØR X=1 TØ L
270 D2=V*(Z/Y)
280 Z=Z-1
290 D3=2*C/L
300 C=C-D3
310 PRINT X,
312 Q=D1
314 GØSUB 400
316 Q=D2
318 GØSUB 400
320 Q=D3
322 GØSUB 400
325 PRINT
330 NEXT X
350 PRINT "FINISHED"
360 GØTØ 999
400 Q=INT(Q*100)/100
420 IF Q>100 THEN 440
430 PRINT " ";
440 IF Q>10 THEN 460
450 PRINT " ";
460 PRINT "$"Q,
490 RETURN
999 END
```

Value to be depreciated = Cost − Scrap Value

Straight Line dep'n = Value / Years

Sum of digits depreciation

Double declining balance depreciation

Print routine rounds off to an even number of dollars or dollars and cents.

```
RUN

ØRIGINAL CØST ØF ITEM (DØLLARS, NØ CENTS)? 3900
LIFE ØF ITEM (YEARS, NØ MØNTHS)? 5
SCRAP VALUE (DØLLARS, NØ CENTS)? 400

YEAR          STRAIGHT       SUM ØF          DØUBLE
                LINE         DIGITS         DECLINING

 1            $ 700          $ 1166.66      $ 1560
 2            $ 700          $ 933.33       $ 936
 3            $ 700          $ 700          $ 561.59
 4            $ 700          $ 466.66       $ 336.95
 5            $ 700          $ 233.33       $ 202.17
FINISHED
```

Systematic Savings

When is it worth taking a simple algebra or finance or engineering formula and writing a computer program around it? Why not simply use an $8 calculator to get the result? Generally that's the best bet except when you want to use the formula over and over again or when amounts are being accumulated from one calculation to the next one.

Here's such an example. This program computes the total amount of money which will accumulate under a systematic investment program. The entire forumla is contained in Line 20; S is the net investment year by year. Use it to find out how much you could save if you weren't investing in a home computer, peripherals, terminals, software, etc., etc. — DHA

```
1 PRINT "THIS PRØGRAM CØMPUTES THE TØTAL AMØUNT ØF MØNEY"
2 PRINT "WHICH WILL ACCUMULATE UNDER A SYSTEMATIC INVESTMENT PRØGRAM"
3 PRINT\PRINT "HØW MUCH DØ YØU WANT TØ INVEST PER YEAR";
4 INPUT A\PRINT "HØW MANY YEARS";\INPUT N
5  PRINT "ANNUAL RATE ØF INTEREST (PERCENT)";\INPUT R
6 PRINT\PRINT "YEAR","TØTAL INVEST","TØTAL ACCUMULATED"
10 FØR N1=1 TØ N
20 S=A*(((1+(R/100))^N1)-1)/(R/100)
30 PRINT N1,N1*A,S
40 NEXT N1
99 END
```

The standard interest accumulation formula seems disguised in Basic code:

$$S = \frac{A[(1+r)^n - 1]}{r}$$

```
READY
RUN

THIS PRØGRAM CØMPUTES THE TØTAL AMØUNT ØF MØNEY
WHICH WILL ACCUMULATE UNDER A SYSTEMATIC INVESTMENT PRØGRAM

HØW MUCH DØ YØU WANT TØ INVEST PER YEAR? 5000
HØW MANY YEARS? 10
ANNUAL RATE ØF INTEREST (PERCENT)? 6.7

YEAR           TØTAL INVEST   TØTAL ACCUMULATED
 1              5000           5000
 2              10000          10335
 3              15000          16027.4
 4              20000          22101.3
 5              25000          28582.1
 6              30000          35497.1
 7              35000          42875.4
 8              40000          50748.
 9              45000          59148.1
 10             50000          68111.1
```

SMART

I know a computer that is so very, very smart.
But it still takes me to tell it when to start.
It can easily compute the orbits of the moons,
Could even play all the Beatle's favorite tunes.
On Napoleon's battle plan it could yet improve.
Could beat Bobby Fisher whatever be his move.
With Einstein's theory it should for sure agree.
But remember, it could not do this, without me.

Edward Stewart

Was a telephone computer so bold
The first of it's kind that was sold.
When you call its number
It is a bummer
It always puts you on hold.

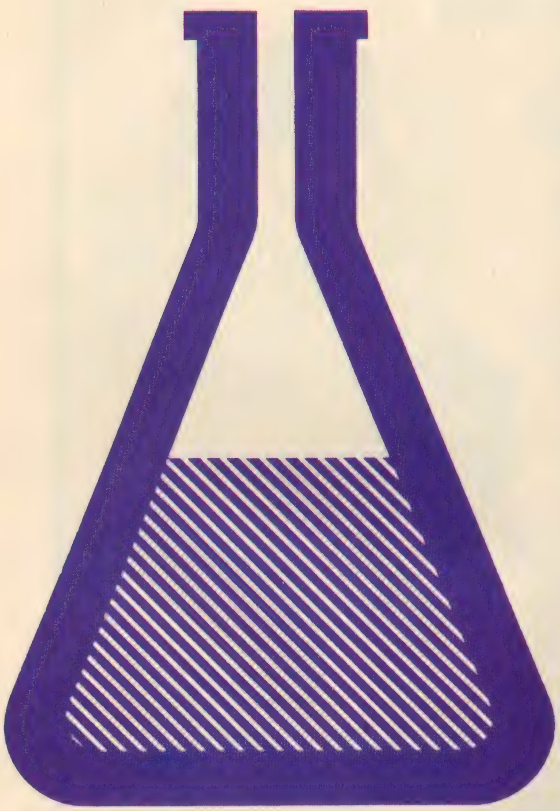
There was a computer from Clyde.
Was indeed so very dignified.
We did not agree
On its pedigree.
It told me to kiss its back side.

There was a computer from South Bend.
That figured income tax without end.
Calculated the tax
A little bit lax.
Who is it they will apprehend?

A student shows how he used a Sol computer to assist other students in their lab work.

Inorganic Chemistry Program

J.P. Peer



Partial Sample Run

```
YOU CAN PICK THE DISPLAY SPEED FOR THIS PROGRAM.
'0' IS STANDARD (FAST DISPLAY), '99' IS VERY SLOW.
Type the display speed ( 0 to 99 ) here:  0


Type the number of Stability Constants ( 1 to 10 ) here:  5


   OKAY.  NOW TYPE IN THE STABILITY CONSTANTS.
Type K( 1 ) here:  1.9E 4
Type K( 2 ) here:  3.5E 3
Type K( 3 ) here:  7.9E 2
Type K( 4 ) here:  1.5E 2
Type K( 5 ) here:  0.28

NOW YOU NEED TO TYPE IN THE ANALYTICAL CONCENTRATIONS
OF THE LIGAND AND METAL ( PLEASE USE MOLES / LITER ).
Type the Ligand concentration ( 0 to 50 ) here:  1.0
Type the Metal concentration ( 0 to 50 ) here:  0.1


ALPHAS                      CONCENTRATIONS OF COMPLEXES
++++++                      +++++++++++++++++++++++++++
# 0 = 9.8625933 E-14  0-Ligand complex = 9.8625933 E-15 molar
# 1 = 1.8738927 E-9   1-Ligand complex = 1.8738927 E-10 molar
# 2 = .00000656       2-Ligand complex = .00000066 molar
# 3 = .00518131       3-Ligand complex = .00051813 molar
# 4 = .77719701       4-Ligand complex = .0777197 molar
# 5 = .21761516       5-Ligand complex = .02176152 molar
```

Dear David Ahl:

I read the following interview with Lee Felsenstein on the Sol computer in the July-August 1977 *Creative Computing.*

Ahl: Do you have some feeling concerning what people are doing with them so far?

Felsenstein: It's very hard for me to tell . . . they're getting SOLs in order to do something. Just what it is we don't know. . . . .

In an effort to clear up some of the mystery I am sending you a sample of what I have been doing with my Sol

Program Listing

```
1000 REM CHM342C MODIFIED FOR SOL.  1977 BY J. P. PEER
1010 REM 417 WEST WATER STREET, BERNE, INDIANA  46711
1020 REM LET'S CLEAR THE CRT SCREEN.  TO DO THIS WE PRINT THE
1030 REM SCREEN-ERASE CHARACTER.  ON SOL IT IS THE 'CLEAR'
1040 REM KEY.  ON OTHER COMPUTERS IT IS THE 'SCREEN ERASE'
1050 REM KEY OR 'CONTROL-L.'  IF YOU ARE USING A HARD-COPY
1060 REM TERMINAL YOU CAN OMIT THIS.
1100 PRINT"(clear key)"
1110 REM SET SPEED OF SOL'S VIDEO DISPLAY MODULE (VDM).  IF
1120 REM YOUR MACHINE IS NOT A SOL YOU CAN OMIT THIS.
1200 SET S=0
1250 DIM K(11),A1(12)
1290 REM A COLON  (:)  SEPARATES STATEMENTS ON ONE LINE.
1300 PRINT:PRINT:PRINT
1400 PRINT"   H I !    T H I S    P R O G R A M    F I N D S    T H E"
1500 PRINT"C O N C E N T R A T I O N S   O F    A L L    S P E C I E S"
1600 PRINT"I N   A    C O M P L E X   I O N    E Q U I L I B R I U M ."
1700 PRINT"Y O U   M U S T   T Y P E   I N   T H E    S T A B I L I T Y"
1800 PRINT"C O N S T A N T S .":PRINT:PRINT:PRINT
1850 PRINT:PRINT:PRINT
1900 REM IF YOU'RE USING A HARD-COPY TERMINAL YOU CAN OMIT THE
1910 REM SUBROUTINE AT LINE 64000.
1920 GOSUB 64000
```

computer. I am a chemistry student and computer fanatic. I bought the Sol because it comes assembled and it has a keyboard instead of those ludicrous toggle switches!

In the Spring of 1976 I began working with Dr. L.A. Bares of Indiana University—Purdue University at Fort Wayne to build a series of interactive programs to assist Inorganic Chemistry students with their laboratory work. This group of programs was written in BASIC on a CDC-6600 and was used this Spring by chemistry students. The series presently has about ten programs. I am sending you one of these programs that I modified to run on my Sol.

This program finds the concentrations of the various species present in a complex-ion equilibrium. The user enters the stability constants and the analytical concentrations of the metal and ligand. The effects of pH and solubility on the equilibrium have been ignored for simplicity.

Our other Inorganic Chemistry programs assist the lab student as he/she finds the stability constants of a complex ion and then, with these constants known, proceed to plot beautiful logarithmic distribution graphs showing the relationship between ligand concentration and various complex ion concentrations.

We also wrote a program to do similar plots of polyprotic acid species versus pH, and a program to plot redox titration curves, just for fun.

The Sol-compatible versions of several of our other programs must wait until Processor Technology produces its long-awaited 8K BASIC. (I've been waiting since February!)

I believe the only non-standard BASIC statement I have used is: SET S = (a number). This changes the video display speed for Processor Technology's VDM. If you aren't using a Processor Technology computer, omit the statements about display speed.

For more information on our programs contact

J. P. Peer
417 West Water Street
Berne, Indiana 46711

and

Dr. L. A. Bares
Indiana University—Purdue University at Fort Wayne
2101 Coliseum Boulevard East
Fort Wayne, Indiana 46805

References

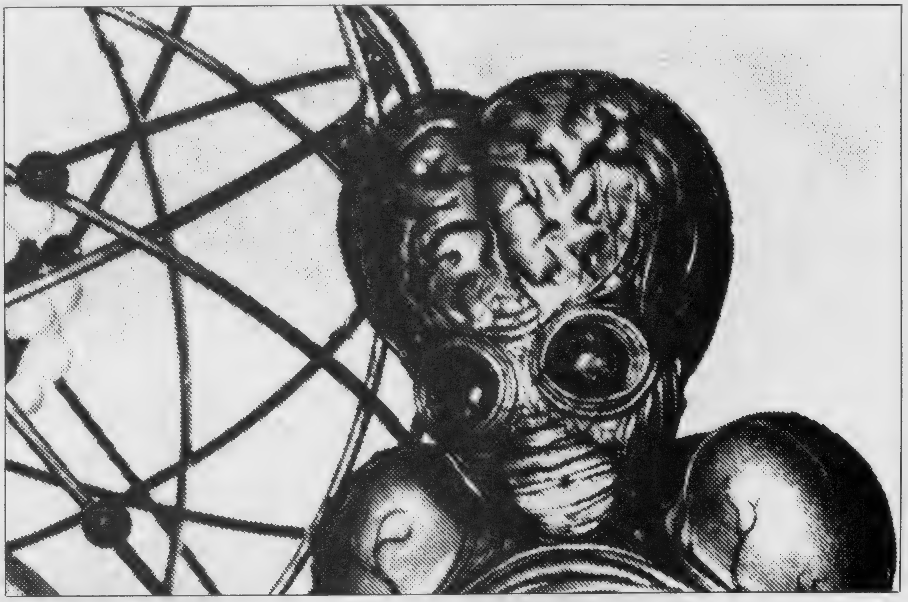
Angelici, R. J., *Synthesis & Technique in Inorganic Chemistry, Experiment* 13 (1977).

Blackburn, T. R., *Equilibrium* (1969).

Li, N. C., J. M. White & R. L. Yoest, *J. Amer. Chem. Soc.*, 78, 5218 (1956).


```
2000 FOR Z=1 TO 6:PRINT:NEXT Z
2100 PRINT"            August, 1977 by J. P. Peer"
2200 FOR Z=1 TO 7:PRINT:NEXT Z
2300 GOSUB 64000
2400 PRINT
2500 L=0:U=99
2600 PRINT"YOU CAN PICK THE DISPLAY SPEED FOR THIS PROGRAM."
2700 PRINT"'0' IS STANDARD (FAST DISPLAY), '99' IS VERY SLOW."
2800 INPUT"Type the display speed ( 0 to 99 ) here: ",S2
2900 A=S2
3000 GOSUB 60000
3050 REM CHANGE VDM DISPLAY SPEED TO USER'S WHIM.
3100 SET S=A
3200 PRINT"(clear key)"
3300 L=1:U=10
3400 PRINT:PRINT
3500 INPUT"Type the number of Stability Constants ( 1 to 10 ) here: ",L1
3510 REM IF YOUR BASIC WON'T ACCEPT THIS TYPE OF INPUT STATEMENT TRY
3520 REM THE FOLLOWING SUBSTITUTION:
3530 REM           PRINT"Type the number of ...... here: ";
3540 REM           INPUT A
3600 A=L1
3700 GOSUB 60000
3800 L1=INT(A)
4000 PRINT"(clear key)"
4100 PRINT:PRINT
4200 PRINT"   OKAY.  NOW TYPE IN THE STABILITY CONSTANTS."
4300 FOR I=1 TO L1
4400 PRINT"Type K(";I;" ) here: "; : INPUT" ",K(I)
4500 NEXT I
4600 PRINT
4700 L=0:U=50
4800 PRINT"NOW YOU NEED TO TYPE IN THE ANALYTICAL CONCENTRATIONS"
4900 PRINT"OF THE LIGAND AND METAL ( PLEASE USE MOLES / LITER )."
5000 INPUT"Type the Ligand concentration ( 0 to 50 ) here: ",L2
5100 A=L2
5200 GOSUB 60000
5300 L2=A
5400 INPUT"Type the Metal concentration ( 0 to 50 ) here: ",M
5500 A=M
5600 GOSUB 60000
5700 M=A
5800 REM FIND alpha 0.  SEE BLACKBURN'S 'EQUILIBRIUM' FOR THE THEORY.
5900 T=1:S=1:N=1
6000 T=T*K(N)*L2
6100 S=S+T
6200 N=N+1
6300 IF N <= L1 THEN 6000
6400 A1(0)=1/S
6500 REM FIND ALL THE OTHER alphas.
6600 N=1
6700 A1(N)=K(N)*A1(N-1)*L2
6800 N=N+1
6900 IF N <=L1 THEN 6700
7000 PRINT"(clear key)"
7100 PRINT
7200 PRINT"ALPHAS";TAB(27);"CONCENTRATIONS OF COMPLEXES"
7300 PRINT"++++++";TAB(27);"++++++++++++++++++++++++++"
7400 FOR J=0 TO L1
7500 PRINT"#";J;" =";A1(J);TAB(21);J;"-Ligand complex =";A1(J)*M;
7550 PRINT" molar"
7600 NEXT J
7650 PRINT
7700 INPUT"To rerun this program type a zero here: ",Z
7800 IF Z ≠ 0 THEN STOP
7900 PRINT"(clear key)"
8000 INPUT"To use the same Stability Constants type zero here: ",Z
8100 IF Z = 0 THEN 4600
8200 GO TO 2400
59999 STOP
60000 REM MY WORLD-FAMOUS ERROR LIMITS SUBROUTINE.  LET'S NOT ALLOW
60010 REM THE USER TO MAKE MISTAKES WHEN S/HE TYPES IN DATA.  INPUT
60020 REM MUST BE WITHIN THE LIMITS OF 'L' AND 'U'.  'A' IS RETURNED.
60030 REM ANY GOOD BASIC INTERPRETER SHOULD HAVE THIS ABILITY BUILT
60040 REM IN AS AN INTRINSIC FUNCTION.
60100 IF A >= L THEN IF A <= U THEN RETURN
60200 PRINT"OOOOPS!  THIS NUMBER MUST BE FROM ";L;" TO ";U
60300 PRINT"PLEASE TRY AGAIN HERE: "; : INPUT" ",A
60400 GO TO 60100
63999 STOP
64000 REM SUBROUTINE TO STOP SCROLLING OF THE SCREEN.  LET'S HAVE
64010 REM PITY ON THE POOR USER WHO CAN'T READ AT 120000 BAUD AND
64020 REM ALLOW HIM/ER TO READ WHAT WE PRINTED BEFORE IT DISAPPEARS.
64030 REM A GOOD INTERPRETER WOULD HAVE THIS AS AN INTRINSIC FUNCTION
64040 REM ALSO.  FOR WORK WITH CRT'S IT'S INEVITABLE.
64100 INPUT"To continue, type any number here: ",Z
64200 PRINT"(clear key)"
64300 RETURN
```

MASTERMIND II



Steve North

In the March-April 1976 issue of *Creative* we published a computerized version of Mastermind, a logic game. Mastermind is played by two people—one is called the code-maker; the other, the code-breaker. At the beginning of the game the code-maker forms a code, or combination of colored pegs. He hides these from the code-breaker. The code-breaker then attempts to deduce the code, by placing his own guesses, one at a time, on the board. After he makes a guess (by placing a combination of colored pegs on the board) the code-maker then gives the code-breaker clues to indicate how close the guess was to the code. For every peg in the guess that's the right color and in the right position, the code-breaker gets a black peg. For every peg in the guess that's the right color but not in the right position, the code-breaker gets a white peg. Note that these black and white pegs do not indicate *which* pegs in the guess are correct, but merely that they exist. For example, if the code was:

Yellow Red Red Green

and my guess was

Red Red Yellow Black

I would receive two white pegs and one black peg for the guess. I wouldn't know (except by comparing previous guesses) which one of the pegs in my guess was the right color in the right position.

Program Listing

```
10 REM
20 REM      MASTERMIND II
30 REM      STEVE NORTH
40 REM      CREATIVE COMPUTING
50 REM      PO BOX 789-M MORRISTOWN NEW JERSEY 07960
60 REM
70 PRINT "MASTERMIND II"
80 INPUT "NUMBER OF COLORS";C9
90 IF C9>8 THEN PRINT "NO MORE THAN 8, PLEASE!":GOTO 80
100 INPUT "NUMBER OF POSITIONS";P9
110 INPUT "NUMBER OF ROUNDS";R9
120 P=C9↑P9
130 PRINT "TOTAL POSSIBILITIES =";P
140 H=0:C=0
150 DIM Q(P9),S(10,2),S$(10),A$(P9),G$(P9),I(P),H$(P9)
160 L$="BWRGOYPT"
170 PRINT
180 PRINT
190 PRINT "COLOR     LETTER"
200 PRINT "=====     ======"
210 FOR X=1 TO C9
220 READ X$
230 PRINT X$;TAB(13);MID$(L$,X,1)
240 NEXT X
250 PRINT
260 FOR R=1 TO R9
270 PRINT
280 PRINT "ROUND NUMBER ";R;"----"
290 PRINT
300 PRINT "GUESS MY COMBINATION."
310 REM      GET A COMBINATION
320 A=INT(P*RND(1)+1)
330 GOSUB 3000
340 FOR X=1 TO A
350 GOSUB 3500
360 NEXT X
370 FOR M=1 TO 10
380 PRINT "MOVE # ";M;" GUESS ";:INPUT X$
390 IF X$="BOARD" THEN 2000
400 IF X$="QUIT" THEN 2500
410 IF LEN(X$)<>P9 THEN PRINT "BAD NUMBER OF POSITIONS.":GOTO 380
420 REM      UNPACK X$ INTO G$(1-P9)
430 FOR X=1 TO P9
440 FOR Y=1 TO C9
450 IF MID$(X$,X,1)=MID$(L$,Y,1) THEN 480
460 NEXT Y
470 PRINT "'"; MID$(X$,X,1); "' IS UNRECOGNIZED.":GOTO 380
480 G$(X)=MID$(X$,X,1)
490 NEXT X
500 REM      NOW WE CONVERT Q(1-P9) INTO A$(1-P9) [ACTUAL GUESS]
510 GOSUB 4000
```

In the version of Mastermind published earlier in *Creative*, the computer could play only the passive role, that of the code-maker. Since then many readers have submitted excellent versions which play both the active and passive roles. Most of the submissions were unfortunately in Fortran so we here present a version of the game written in MITS 8K BASIC.

Actually, the task of getting the computer to deduce the correct combination is not at all difficult. Imagine, for instance, that you made a list of all the possible codes. To begin, you select a guess from your list at random. Then, as you receive clues, you cross off from the list those combinations which you know are impossible. For example if your guess is Red Red Green Green and you receive no pegs, then you know that any combination containing either a red or a green peg is impossible and may be crossed off the list. The process is continued until the correct solution is reached or there are no more combinations left on the list (in which case you know that the code-maker make a mistake in giving you the clues somewhere).

Note that in this particular implementation, we never actually create a list of the combinations, but merely keep track of which ones (in sequential order) may be correct. Using this system, we can easily say that the 523rd combination may be correct, but to actually produce the 523rd combination we have to count all the way from the first combination (or the previous one, if it was lower than 523). Actually, this problem could be simplified to a conversion from base 10 to base (number-of-colors) and then adjusting the values used in the MID$ function so as not to take a zeroth character from a string if you want to experiment. We did try a version that kept an actual list of all possible combinations (as a string array), which was significantly faster than this version, but which ate tremendous amounts of memory.

At the beginning of this game, you input the number of colors and number of positions you wish to use (which will directly affect the number of combinations) and the number of rounds you wish to play. While you are playing as the code-breaker, you may type in BOARD at any time to get a list of your previous guesses and clues, and QUIT to end the game. Note that this version uses string arrays, but this is merely for convenience and can easily be converted for a BASIC that has no string arrays as long as it has a MID$ function. This is because the string arrays are one-dimensional, never exceed a length greater than the number of positions, and the elements never contain more than one character.

```
520 REM     AND GET NUMBER OF BLACKS AND WHITES
530 GOSUB 4500
540 IF B=P9 THEN 630
550 REM     TELL HUMAN RESULTS
560 PRINT "YOU HAVE ";B;" BLACKS AND ";W;" WHITES."
570 REM     SAVE ALL THIS STUFF FOR BOARD PRINTOUT LATER
580 SS(M)=XS
590 S(M,1)=B
600 S(M,2)=W
610 NEXT M
620 PRINT "YOU RAN OUT OF MOVES!  THAT'S ALL YOU GET!":GOTO 640
622 GOSUB 4000
623 PRINT "THE ACTUAL COMBINATION WAS: ";
624 FOR X=1 TO P9
625 PRINT A$(X);
626 NEXT X
627 PRINT
630 PRINT "YOU GUESSED IT IN ";M;" MOVES!"
640 H=H+M
650 GOSUB 5000
660 REM
670 REM     NOW COMPUTER GUESSES
680 REM
690 FOR X=1 TO P
700 I(X)=1
710 NEXT X
720 PRINT "NOW I GUESS.  THINK OF A COMBINATION."
730 INPUT "HIT RETURN WHEN READY ";X$
740 FOR M=1 TO 10
750 GOSUB 3000
760 REM     FIND A GUESS
770 G=INT(P*RND(1)+1)
780 IF I(G)=1 THEN 890
790 FOR X=G TO P
800 IF I(X)=1 THEN 880
810 NEXT X
820 FOR X=1 TO G
830 IF I(X)=1 THEN 880
840 NEXT X
850 PRINT "YOU DUMMY, YOU HAVE GIVEN ME INCONSISTENT INFORMATION."
860 PRINT "LET'S TRY AGAIN, AND THIS TIME, BE MORE CAREFUL."
870 GOTO 660
880 G=X
890 REM     NOW WE CONVERT GUESS #G INTO G$
900 FOR X=1 TO G
910 GOSUB 3500
920 NEXT X
930 GOSUB 6000
940 PRINT "MY GUESS IS: ";
950 FOR X=1 TO P9
960 PRINT H$(X);
970 NEXT X
980 INPUT "  BLACKS, WHITES ";B1,W1
990 IF B1=P9 THEN 1120
1000 GOSUB 3000
1010 FOR X=1 TO P
1020 GOSUB 3500
1030 IF I(X)=0 THEN 1070
1035 GOSUB 6500
1040 GOSUB 4000
1050 GOSUB 4500
1060 IF B1<>B OR W1<>W THEN I(X)=0
1070 NEXT X
1080 NEXT M
1090 PRINT "I USED UP ALL MY MOVES!"
1100 PRINT "I GUESS MY CPU IS JUST HAVING AN OFF DAY."
1110 GOTO 1130
1120 PRINT "I GOT IT IN ";M;" MOVES!"
1130 C=C+M
1140 GOSUB 5000
1150 NEXT R
1160 PRINT "GAME OVER"
1170 PRINT "FINAL SCORE:"
1180 GOSUB 5040
1190 STOP
2000 REM
2010 REM     BOARD PRINTOUT ROUTINE
2020 REM
2025 PRINT
2030 PRINT "BOARD"
2040 PRINT "MOVE     GUESS          BLACK     WHITE"
2050 FOR Z=1 TO M-1
2060 PRINT Z;TAB(9);S$(Z);TAB(25);S(Z,1);TAB(35);S(Z,2)
2070 NEXT Z
2075 PRINT
2080 GOTO 380
2500 REM
2510 REM     QUIT ROUTINE
2520 REM
```

```
2530 PRINT "QUITTER!  MY COMBINATION WAS: ";
2535 GOSUB 4000
2540 FOR X=1 TO P9
2550 PRINT A$(X);
2560 NEXT X
2565 PRINT
2570 PRINT "GOOD BYE"
2580 STOP
3000 REM
3010 REM      INITIALIZE Q(1-P9) TO ZEROS
3020 REM
3030 FOR S=1 TO P9
3040 Q(S)=0
3050 NEXT S
3060 RETURN
3500 REM
3510 REM      INCREMENT Q(1-P9)
3520 REM
3522 IF Q(1)>0 THEN 3530
3524 REM  IF ZERO, THIS IS OUR FIRST INCREMENT: MAKE ALL ONES
3526 FOR S=1 TO P9
3527 Q(S)=1
3528 NEXT S
3529 RETURN
3530 Q=1
3540 Q(Q)=Q(Q)+1
3550 IF Q(Q)<=C9 THEN RETURN
3560 Q(Q)=1
3570 Q=Q+1
3580 GOTO 3540
4000 REM
4010 REM      CONVERT Q(1-P9) TO A$(1-P9)
4020 REM
4030 FOR S=1 TO P9
4040 A$(S)=MID$(L$,Q(S),1)
4050 NEXT S
4060 RETURN
4500 REM
4510 REM      GET NUMBER OF BLACKS (B) AND WHITES (W)
4520 REM      MASHES G$ AND A$ IN THE PROCESS
4530 REM
4540 B=0:W=0:F=0
4550 FOR S=1 TO P9
4560 IF G$(S)<>A$(S) THEN 4620
4570 B=B+1
4580 G$(S)=CHR$(F)
4590 A$(S)=CHR$(F+1)
4600 F=F+2
4610 GOTO 4660
4620 FOR T=1 TO P9
4630 IF G$(S)<>A$(T) THEN 4650
4640 IF G$(T)=A$(T) THEN 4650
4645 W=W+1:A$(T)=CHR$(F):G$(S)=CHR$(F+1):F=F+2:GOTO 4660
4650 NEXT T
4660 NEXT S
4670 RETURN
5000 REM
5010 REM      PRINT SCORE
5020 REM
5030 PRINT "SCORE:"
5040 PRINT "      COMPUTER ";C
5050 PRINT "      HUMAN    ";H
5060 PRINT
5070 RETURN
5500 REM
5510 REM      CONVERT Q(1-P9) INTO G$(1-P9)
5520 REM
5530 FOR S=1 TO P9
5540 G$(S)=MID$(L$,Q(S),1)
5550 NEXT S
5560 RETURN
6000 REM
6010 REM      CONVERT Q(1-P9) TO H$(1-P9)
6020 REM
6030 FOR S=1 TO P9
6040 H$(S)=MID$(L$,Q(S),1)
6050 NEXT S
6060 RETURN
6500 REM
6510 REM      COPY H$ INTO G$
6520 REM
6530 FOR S=1 TO P9
6540 G$(S)=H$(S)
6550 NEXT S
6560 RETURN
8000 REM      PROGRAM DATA FOR COLOR NAMES
8010 DATA BLACK,WHITE,RED,GREEN,ORANGE,YELLOW,PURPLE,TAN
9998 REM   ...WE'RE SORRY BUT IT'S TIME TO GO...
9999 END
```

Sample Run

```
RUN
MASTERMIND II
NUMBER OF COLORS? 4
NUMBER OF POSITIONS? 4
NUMBER OF ROUNDS? 1
TOTAL POSSIBILITIES = 256


COLOR      LETTER
=====      ======
BLACK        B
WHITE        W
RED          R
GREEN        G


ROUND NUMBER  1 ----

GUESS MY COMBINATION.
MOVE #  1   GUESS ? BWRG
YOU HAVE  2  BLACKS AND  1  WHITES.
MOVE #  2   GUESS ? BWGG
YOU HAVE  1  BLACKS AND  1  WHITES.
MOVE #  3   GUESS ? BRRW
YOU HAVE  3  BLACKS AND  0  WHITES.
MOVE #  4   GUESS ? BRRB
YOU HAVE  2  BLACKS AND  1  WHITES.
MOVE #  5   GUESS ? BBRW
YOU GUESSED IT IN  5  MOVES!
SCORE:
     COMPUTER  0
     HUMAN     5

NOW I GUESS.  THINK OF A COMBINATION.
HIT RETURN WHEN READY ?
MY GUESS IS: GRGR  BLACKS, WHITES ? 1,1
MY GUESS IS: GBRB  BLACKS, WHITES ? 1,0
MY GUESS IS: WRRW  BLACKS, WHITES ? 0,1
MY GUESS IS: GGWG  BLACKS, WHITES ? 4,0
I GOT IT IN  4  MOVES!
SCORE:
     COMPUTER  4
     HUMAN     5
```

```
RUN
MASTERMIND II
NUMBER OF COLORS? 4
NUMBER OF POSITIONS? 5
NUMBER OF ROUNDS? 2
TOTAL POSSIBILITIES = 1024


COLOR      LETTER
=====      ======
BLACK        B
WHITE        W
RED          R
GREEN        G


ROUND NUMBER  1 ----

GUESS MY COMBINATION.
MOVE #  1   GUESS ? BBWWG
YOU HAVE  0  BLACKS AND  3  WHITES.
MOVE #  2   GUESS ? WWBRR
YOU HAVE  0  BLACKS AND  3  WHITES.
MOVE #  3   GUESS ? GGRBW
YOU HAVE  3  BLACKS AND  0  WHITES.
MOVE #  4   GUESS ? GRRBB
YOU GUESSED IT IN  4  MOVES!
SCORE:
     COMPUTER  0
     HUMAN     4

NOW I GUESS.  THINK OF A COMBINATION.
HIT RETURN WHEN READY ?
MY GUESS IS: BBWWR  BLACKS, WHITES ? 1,2
MY GUESS IS: WWRGR  BLACKS, WHITES ? 1,1
MY GUESS IS: RBBGG  BLACKS, WHITES ? 1,4
MY GUESS IS: BRGGB  BLACKS, WHITES ? 0,5
MY GUESS IS: GGBBR  BLACKS, WHITES ? 5,0
I GOT IT IN  5  MOVES!
SCORE:
     COMPUTER  5
     HUMAN     4
```

OTHELLO

Ed Wright*

Language: FORTRAN
Description: Othello is a two-player game played on an 8-by-8 grid. The game begins with four chips on the board—two black and two white. In alternating moves, the players place a chip on the board (one player uses white chips, the other black chips). When, by making your move, you succeed in getting one or more chips of the opposing color between two of yours (in a horizontal, vertical, or diagonal row) they are converted to chips of your color. The game ends when the board is completely filled and the winner is determined on the basis of who has more chips of his color on the board. In the computerized version of the game presented here, you are pitted against the computer.

The author provided the following explanation of each subroutine:

Subroutine MOVEG (B, OC, NM, MOVESI, MOVESJ, DIR, LC, JAA, IAA IM, NØMVE, NFLIP): Generates all the legal moves from any possible board position. B is a 2D array that stores the present board position (100="off the board," 1=white, -1=black, 0=empty square). ØC is the "opposite-color variable" and tells the program what color the opponent is. If - ØC is put in the place of ØC, then the subroutine will generate all the legal moves for the opponent. MOVESI and MOVESJ are arrays that store the I and J coordinates of each of the legal moves returned from the subroutine. DIR is a 2D array that stores the number of the direction that pieces are flipped for each move. LC is an array that stores how many different directions pieces are flipped for each move. IM is the number of possible moves generated. NØMVE (could be eliminated and IM used) is a variable that if equal to 1 indicates no moves exist for the player from the present board position. NØMVE equal to 0 indicates the player does have at least one move. NFLIP is an array that stores how many pieces are flipped for each of the IM moves.

Subroutine CØUNT(B, ØC, NØC): Counts up the opponent's number of pieces on the board. If -ØC is put in place of ØC, then it counts up the number of pieces the computer has.

Subroutine BOARDP(B,NM): Prints out the board after having converted the 1's and 0's and -1's to W's, .'s, and B's.(See sample game included).

Subroutine BOARDC(MOVESI, MOVESJ, IF, IAA, JAA, B, ØC, DIR, LC): Changes the board for the IFth move of the possible moves stored in MOVESI, MOVESJ. Depending upon ØC, it too can either change the board for a move made by the computer or by its opponent.

Subroutine MOVEE(B, ØC, NM, MOVESI, MOVESJ, NFLIP, DIR, LC, IM, IF, IAA, JAA): Evaluates each of the moves given to it from the move-generator subroutine and returns what it considers to be the best move via the integer IF; that is, the move it selects is MOVESI(IF), MOVESJ(IF). I have not tried it, but I believe you could replace ØC with -ØC in this routine and set up the program to play itself and "fine-tune" the evaluation of the moves. The present version of MOVEE only looks ahead two ply and evaluates all the possible board positions. I am working on a version of MOVEE that will look ahead four ply (two turns) and use pruning techniques and a more general evaluation "function."

Final note: Originally, if the computer was allowed to go first, I had a random-number generator that chose one of the four possible opening moves for the computer to make. However, since the opening position is completely symmetric, it makes no difference which of these four opening moves one makes. Therefore, in the interest of reducing slightly the size of the program, I removed the random-number generator and associated statements and now just allow the program to make one of the four possible opening moves. I feel the savings in space is more important to the computer hobbyist.

Also, I am continually fine-tuning the move-evaluation subroutine that is included the printout, although its basic structure will not change.

Sample Run

—Game opening—

```
          BOARD POSITION AFTER    Ø MOVES

J = 1  2  3  4  5  6  7  8
I
"

1   .  .  .  .  .  .  .  .
2   .  .  .  .  .  .  .  .
3   .  .  .  .  .  .  .  .
4   .  .  .  W  B  .  .  .
5   .  .  .  B  W  .  .  .
6   .  .  .  .  .  .  .  .
7   .  .  .  .  .  .  .  .
8   .  .  .  .  .  .  .  .


WELCOME TO THE GAME OF OTHELLO. DO YOU WISH TO
GO FIRST? YOU ARE WHITE IF YOU ARE FIRST.
?
'yes'

DO YOU WISH TO BE GIVEN A HANDICAP?
?
'no'

DO YOU WISH TO GIVE ME A HANDICAP?
?
'no'


WHAT IS YOUR MOVE ? (I,J).
? ØØØ14
4,6

          BOARD POSITION AFTER    1 MOVES

J = 1  2  3  4  5  6  7  8
I
"

1   .  .  .  .  .  .  .  .
2   .  .  .  .  .  .  .  .
3   .  .  .  .  .  .  .  .
4   .  .  .  W  W  W  .  .
5   .  .  .  B  W  .  .  .
6   .  .  .  .  .  .  .  .
7   .  .  .  .  .  .  .  .
8   .  .  .  .  .  .  .  .

MY MOVE IS : 5,6
```

*802 Coriander Dr., Torrance, CA 90502.

—Later in the game—

```
WHAT IS YOUR MOVE ? (I,J).
? 00014
3,5

     BOARD POSITION AFTER  11 MOVES

J = 1  2  3  4  5  6  7  8
I
"
1   .  .  .  .  .  .  .  .
2   .  .  .  .  .  .  .  .
3   .  .  .  .  W  B  .  .
4   .  .  .  W  W  W  .  .
5   B  B  B  B  W  W  .  .
6   .  .  .  W  W  W  .  .
7   .  .  .  W  .  .  .  .
8   .  .  .  .  .  .  .  .

MY MOVE IS : 8,4

     BOARD POSITION AFTER  12 MOVES
```

—Still later in the game—

```
WHAT IS YOUR MOVE ? (I,J).
? 00014
3,8

     BOARD POSITION AFTER  23 MOVES

J = 1  2  3  4  5  6  7  8
I
"
1   .  .  .  .  .  .  .  .
2   .  .  W  .  .  .  .  .
3   .  .  W  W  W  W  W  W
4   .  .  W  B  W  B  W  W
5   B  B  W  B  B  W  B  .
6   .  .  W  B  W  W  .  B
7   .  .  .  B  .  .  .  .
8   .  .  .  B  .  .  .  .

MY MOVE IS : 2,4

     BOARD POSITION AFTER  24 MOVES
```

—At the end of the game—

```
WHAT IS YOUR MOVE ? (I,J).
? 00014
7,1

     BOARD POSITION AFTER  59 MOVES

J = 1  2  3  4  5  6  7  8
I
"
1   B  B  B  B  B  B  B  B
2   W  B  B  B  W  W  .  B
3   W  B  B  B  W  W  B  B
4   W  B  B  B  W  B  W  B
5   W  B  W  W  B  W  W  B
6   W  W  W  W  W  B  W  B
7   W  W  W  W  W  W  W  B
8   B  B  B  B  B  B  B  B

MY MOVE IS : 2,7

     BOARD POSITION AFTER  60 MOVES
```

Program Listing

```
      INTEGER B(10,10),DIR(30,8)
      DIMENSION DRSPON(2),IAA(8),JAA(8),MOVESI(30)
     2,MOVESJ(30),LC(30),NFLIP(30)
      DATA DRSPON /'YES','NO'/
      DATA IAA /-1,-1,-1,0,1,1,1,0/
      DATA JAA /-1,0,1,1,1,0,-1,-1/
      COMMON /TWO/ OC
   22 DO 10 I=1,10
      DO 10 J=1,10
      B(I,J)=0
      IF(I.EQ.1.OR.I.EQ.10)B(I,J)=100
   10 IF(J.EQ.1.OR.J.EQ.10)B(I,J)=100
      B(5,5)=1
      B(5,6)=-1
      B(6,5)=-1
      B(6,6)=1
      CALL BOARDP(B,0,0)
      WRITE(6,601)
  601 FORMAT(//,1X,'WELCOME TO THE GAME OF OTHELLO. DO YOU WISH TO
     2',/,' GO FIRST? YOU ARE WHITE IF YOU ARE FIRST.')
      READ(5,*)RESPON
      OC=1
      IF(RESPON.EQ.DRSPON(2))GO TO 11
      CALL HANDIC(OC,B,DRSPON,NHD)
      NM=NHD
    8 IF(NM.EQ.60)GO TO 15
      CALL MOVEG(B,-OC,NM,MOVESI,MOVESJ,DIR,LC,JAA,IAA,
     2IM,NOMVE,NFLIP)
      IF(IM.EQ.0)GO TO 12
      WRITE(6,713)
  713 FORMAT(//,1X,'WHAT IS YOUR MOVE ? (I,J).')
   14 READ(5,*)MOVEI,MOVEJ
      MOVEJ=MOVEJ+1
      MOVEI=MOVEI+1
      DO 9 I=1,IM
      IF(MOVESI(I).EQ.MOVEI.AND.MOVESJ(I).EQ.MOVEJ)GO TO 13
    9 CONTINUE
      WRITE(6,701)
  701 FORMAT(1X,'MOVE INVALID. PLEASE RE-ENTER')
      GO TO 14
   13 NM=NM+1
      CALL BOARDC(MOVESI,MOVESJ,I,IAA,JAA,B,-OC,DIR,LC)
      CALL BOARDP(B,NM,NHD)
      GO TO 2
   11 OC=-1
      CALL HANDIC(OC,B,DRSPON,NHD)
      B(5,7)=1
      B(5,6)=1
      NM=NHD+1
      CALL BOARDP(B,NM,NHD)
      GO TO 8
   12 WRITE(6,756)
  756 FORMAT(/,1X,'I CAN SEE NO MOVE FOR YOU , SO I WILL ',
     2'MOVE IF I CAN.')
    2 IF(NM.EQ.60)GO TO 15
      CALL MOVEG(B,OC,NM,MOVESI,MOVESJ,DIR,LC,JAA,IAA,
     2IM,NOMVE,NFLIP)
      IF(IM.EQ.0)GO TO 20
      CALL MOVEE(B,OC,NM,MOVESI,MOVESJ,NFLIP,DIR,LC,IM,IF,IAA,JAA)
      MOVEI=MOVESI(IF)-1
      MOVEJ=MOVESJ(IF)-1
      WRITE(6,603)MOVEI,MOVEJ
  603 FORMAT(/,1X,'MY MOVE IS : ',I1,',',I1)
      CALL BOARDC(MOVESI,MOVESJ,IF,IAA,JAA,B,OC,DIR,LC)
      NM=NM+1
      CALL BOARDP(B,NM,NHD)
      GO TO 8
   20 WRITE(6,602)
  602 FORMAT(/,1X,'DO YOU HAVE A MOVE?')
      READ(5,*)RESPON
      IF(RESPON.EQ.DRSPON(1))GO TO 8
      IF(IM.NE.0)GO TO 2
   15 CALL COUNT(B,OC,NOC)
      CALL COUNT(B,-OC,NC)
      IF(NOC.GT.NC)WRITE(6,610)
  610 FORMAT(/,1X,'CONGRATULATIONS,YOU PLAYED WELL AND HAVE WON.',
     2/,1X,'THANK YOU FOR A FINE GAME')
      IF(NOC .LT. NC)WRITE(6,611)
  611 FORMAT(/,1X,'YOU PLAYED WELL ; HOWEVER,YOUR LUCK WAS BAD AND'

     2,/,'I HAVE WON. THANK YOU FOR A FINE GAME.')
      IF(NOC .EQ. NC)WRITE(6,612)
  612 FORMAT(/,'YOU PLAYED WELL AND WE HAVE TIED. I WAS LUCKY.',/,
     2'THANK YOU FOR A FINE GAME.')
      WRITE(6,613)
  613 FORMAT(/,1X,'DO YOU WISH TO PLAY AGAIN?')
      READ(5,*)RESPON
      IF(RESPON.EQ.DRSPON(1))GO TO 22
      STOP
      END
      SUBROUTINE MOVEG(B,OC,NM,MOVESI,MOVESJ,DIR,LC,JAA,IAA,IM,
     2NOMVE,NFLIP)
      INTEGER B(10,10),DIR(30,8)
      DIMENSION MOVESI(30),MOVESJ(30),LC(30),NFLIP(30),
     2IAA(1),JAA(1)
      COMMON /TWO/ OCA
      DO 1 I=1,30
      LC(I)=0
    1 NFLIP(I)=0
      IM=0
      DO 20 I=2,9
      DO 20 J=2,9
      IF(B(I,J))20,3,20
    3 IC=0
      DO 5 L=1,8
      IA=IAA(L)
      JA=JAA(L)
      IF(B(I+IA,J+JA)-OC)5,6,5
    6 IV=1
    4 IV=IV+1
      MVI=I+IV*IA
      MVJ=J+IV*JA
      IF(B(MVI,MVJ))7,5,7
    7 IF(B(MVI,MVJ)-100)11,5,11
   11 IF(B(MVI,MVJ)-OC)8,4,8
    8 IF(IC-1)13,12,13
   13 IM=IM+1
      IC=1
   12 NFLIP(IM)=NFLIP(IM)+IV
      LC(IM)=LC(IM)+1
      LD=LC(IM)
      DIR(IM,LD)=L
    5 CONTINUE
      IF(IC)21,20,21
   21 MOVESI(IM)=I
      MOVESJ(IM)=J
   20 CONTINUE
      IF(IM)31,31,30
   31 IF(OCA.EQ.OC)WRITE(6,100)
  100 FORMAT(/,1X,'I HAVE NO MOVE AND MUST PASS')
   30 RETURN
      END
      SUBROUTINE COUNT(B,OC,NOC)
      INTEGER B(10,1)
      NOC=0
      DO 10 I=2,9
      DO 10 J=2,9
      IF(B(I,J)-OC)10,5,10
    5 NOC=NOC+1
   10 CONTINUE
      RETURN
      END
      SUBROUTINE BOARDP(B,NM,NHD)
      DIMENSION OUT(3),POUT(10,10)
      INTEGER B(10,1)
      DATA OUT /'B','.','W'/
      NMP=NM-NHD
      WRITE(6,100)NMP
  100 FORMAT(/,6X,'BOARD POSITION AFTER  ',I2,' MOVES',/)
      WRITE(6,101)
  101 FORMAT(' J = 1  2  3  4  5  6  7  8')
      WRITE(6,102)
  102 FORMAT(' I')
      WRITE(6,103)
  103 FORMAT(' "')
      DO 9 I=2,9
      DO 9 J=2,9
      IS=B(I,J)+2
    9 POUT(I,J)=OUT(IS)
      DO 10 I=2,9
      I1=I-1
   10 WRITE(6,104)I1,(POUT(I,J),J=2,9)
  104 FORMAT(1X,I1,3X,8(A1,2X))
```

```
      RETURN
      END
      SUBROUTINE BOARDC(MOVESI,MOVESJ,IF,IAA,JAA,B,OC,DIR,LC)
      INTEGER B(10,10),DIR(30,8)
      DIMENSION MOVESI(30),MOVESJ(30),IAA(1),JAA(1),LC(30)
      MI=MOVESI(IF)
      MJ=MOVESJ(IF)
      B(MI,MJ)=-OC
      NDIR=LC(IF)
      DO 40 I=1,NDIR
      L=DIR(IF,I)
      IA=IAA(L)
      JA=JAA(L)
      IV=0
   31 IV=IV+1
      MVI=MI+IV*IA
      MVJ=MJ+IV*JA
      IF(B(MVI,MVJ)+OC)39,40,39
   39 B(MVI,MVJ)=-OC
      GO TO 31
   40 CONTINUE
      RETURN
      END
      SUBROUTINE MOVEE(B,OC,NM,MOVESI,MOVESJ,NFLIP,DIR,LC,
     2IM,IF,IAA,JAA)
      INTEGER B(10,1),DIR(30,1),BT(10,10),BTT(10,10),DIRB(20,8)
     2,BTTS(9,9,20),DIRBB(20,8)
      DIMENSION MOVESI(1),MOVESJ(1),LC(1),NFLIP(1),MBI(20),MBJ(20)
     2,LCB(20),NFLIPB(30),IAA(1),JAA(1),IY(24),JY(24),
     2IMID(24),JMID(24),ID(24),JD(24),NCORNI(4),NCORNJ(4)
     2,MBBI(20),MBBJ(20),LCBB(20),NFLIB(30)
      DATA NCORNI,NCORNJ /2,2,9,9,2,9,9,2/
      DATA ID,JD /3,4,5,6,7,8,6*9,8,7,6,5,4,3,12*2,3,4,5,6,7,8,
     26*9,8,7,6,5,4,3/
      DATA IY,JY /5,1,3,8,1,6,9,1,9,9,1,9,6,1,8,3,1,5,2,1,2,2,1,2,
     22,1,2,2,1,2,5,1,3,8,1,6,9,1,9,9,1,9,6,1,8,3,1,5/
      DATA IMID,JMID /4,1,4,7,1,7,9,1,9,9,1,9,7,1,7,4,1,4,2,1,2,2,
     21,2,2,1,2,2,1,2,4,1,4,7,1,7,9,1,9,9,1,9,7,1,7,4,1,4/
      ICO=0
      IF=1
      IF(NM.EQ.59)GO TO 20
   10 DO 12 I=1,IM
      MI=MOVESI(I)
      MJ=MOVESJ(I)
      IF(MI.NE.3.AND.MI.NE.8)GO TO 13
      IF(MJ.NE.3.AND.MJ.NE.8)GO TO 13
      IF(MI.EQ.3.AND.MJ.EQ.3)IC=1
      IF(MI.EQ.3.AND.MJ.EQ.8)IC=2
      IF(MI.EQ.8.AND.MJ.EQ.8)IC=3
      IF(MI.EQ.8.AND.MJ.EQ.3)IC=4
      IF(B(NCORNI(IC),NCORNJ(IC)).EQ.0)NFLIP(I)=NFLIP(I)-50
   13 IF(MI.NE.2.AND.MI.NE.9)GO TO 11
      IF(MJ.NE.2.AND.MJ.NE.9)GO TO 11
      ICO=ICO+1
      NFLIP(I)=NFLIP(I)+60
   11 IF(MI.LE.3.OR.MI.GE.8)GO TO 2
      IF(MJ.LE.3.OR.MJ.GE.8)GO TO 2
      NFLIP(I)=NFLIP(I)+10
      GO TO 12
    2 ND=LC(I)
      DO 5 J=1,ND
      L=DIR(I,J)
      IA=IAA(L)
      JA=JAA(L)
      IV=1
    4 IV=IV+1
      MVI=MI+IV*IA
      MVJ=MJ+IV*JA
      IF(B(MVI,MVJ)-OC)6,4,6
    6 IV=IV+1
      MVI=MI+IV*IA
      MVJ=MJ+IV*JA
      IF(B(MVI,MVJ)-OC)7,8,7
    7 IF(B(MVI,MVJ)+OC)5,6,5
    8 IF(B(MI-IA,MJ-JA))5,9,5
    9 NFLIP(I)=NFLIP(I)-5
      GO TO 12
    5 CONTINUE
   12 CONTINUE
      DO 32 I=1,IM
      NSUBO=0
      MI=MOVESI(I)
      MJ=MOVESJ(I)
      IC=0
      DO 33 K=1,10
      DO 33 J=1,10
   33 BT(K,J)=B(K,J)
      LL=0
      DO 56 J=1,24
      IPP=ID(J)
      JPP=JD(J)
      IF(MOVESI(I).NE.IPP.OR.MOVESJ(I).NE.JPP)GO TO 56
      LL=J
   56 CONTINUE
      CALL BOARDC(MOVESI,MOVESJ,I,IAA,JAA,BT,OC,DIR,LC)
      CALL MOVEG(BT,-OC,NM,MBI,MBJ,DIRB,LCB,JAA,IAA,IM1,
     2NOMVE,NFLIPB)
      IF(IM1.NE.0)GO TO 63
      NFLIP(I)=NFLIP(I)+100
      GO TO 32
   63 DO 36 J=1,IM1
      DO 34 K=1,10
      DO 34 L=1,10
   34 BTT(K,L)=BT(K,L)
      CALL BOARDC(MBI,MBJ,J,IAA,JAA,BTT,-OC,DIRB,LCB)
      IF(LL.EQ.0)GO TO 38
      IC=1
      IZ=IY(LL)
      JZ=JY(LL)
      IF(B(IZ,JZ)+OC)41,40,41
   40 MK=JMID(LL)
      ML=IMID(LL)
      IF(B(ML,MK).EQ.0)NSUBO=90
   41 IF(BTT(MI,MJ).NE.OC)GO TO 38
      NFLIP(I)=NFLIP(I)-40
      IC=2
   38 CONTINUE
      CALL COUNT(BTT,-OC,NOC)
      IF(NOC)39,39,42
   39 NFLIP(I)=NFLIP(I)-200
      GO TO 32
   42 DO 37 K1=2,9
      DO 37 K2=2,9
   37 BTTS(K1,K2,J)=BTT(K1,K2)
      DO 100 IL=2,9
      DO 100 JL=2,9
      IF(BTT(IL,JL))96,100,96
   96 IF(BTT(IL,JL) - OC)99,100,99
   99 DO 90 IZ=1,8
      IV=0
   80 IV=IV+1
      ILL=IL+IV*IAA(IZ)
      JLL=JL+IV*JAA(IZ)
      IF(BTT(ILL,JLL))98,36,98
   98 IF(BTT(ILL,JLL)-100)97,36,97
   97 IF(BTT(ILL,JLL)-OC)80,90,80
   90 CONTINUE
  100 CONTINUE
   95 CALL MOVEG(BTT,OC,NM,MBBI,MBBJ,DIRBB,LCBB,JAA,IAA,IM2,
     2NOMVE,NFLIB)
      IF(IM2.EQ.0)GO TO 103
      DO 102 IL=1,IM2
      IF(MBBI(IL).NE.2.OR.MBBI(IL).NE.9)GO TO 102
      IF(MBBJ(IL).NE.2.OR.MBBJ(IL).NE.9)GO TO 102
      GO TO 36
  102 CONTINUE
  103 NFLIP(I)=NFLIP(I)-190
   36 CONTINUE
      IF(IC.NE.1)GO TO 35
      DO 50 K=1,24
      IQ=ID(K)
      JQ=JD(K)
      IF(MI.EQ.IQ.AND.MJ.EQ.JQ)GO TO 50
   52 IF(B(IQ,JQ)+OC)50,53,50
   53 DO 54 K1=1,IM1
   54 IF(BTTS(IQ,JQ,K1).EQ.OC)NFLIP(I)=NFLIP(I)-8
   50 CONTINUE
      NFLIP(I)=NFLIP(I)+25-NSUBO
   35 DO 60 K=1,4
      KC1=NCORNI(K)
      KC2=NCORNJ(K)
      IF(B(KC1,KC2))60,58,60
   58 DO 61 K1=1,IM1
   61 IF(BTTS(KC1,KC2,K1).EQ.OC)NFLIP(I)=NFLIP(I)-55
      IF(ICO.LE.1)GO TO 60
      IF(MI.EQ.KC1.AND.MJ.EQ.KC2)GO TO 66
      DO 62 K1=1,IM1
   62 IF(BTTS(KC1,KC2,K1).EQ.OC)NFLIP(I)=NFLIP(I)-20
   60 CONTINUE
   32 CONTINUE
      NFLIPM=-800
      DO 15 I=1,IM
      IF(NFLIP(I)-NFLIPM)15,16,16
   16 NFLIPM=NFLIP(I)
      IF=I
   15 CONTINUE
   20 RETURN
      END
      SUBROUTINE HANDIC(OC,B,DRSPON,NHD)
      DIMENSION DRSPON(1)
      INTEGER B(10,1)
      NHD=0
      WRITE(6,608)
  608 FORMAT(/,1X,'DO YOU WISH TO BE GIVEN A HANDICAP?')
      READ(5,*)RESPON
      IF(RESPON.EQ.DRSPON(1))GO TO 7
      WRITE(6,610)
  610 FORMAT(/,1X,'DO YOU WISH TO GIVE ME A HANDICAP?')
      READ(5,*)RESPON
      IF(RESPON.EQ.DRSPON(2))GO TO 100
      NAH=-OC
      WRITE(6,609)
  609 FORMAT(/,1X,'HOW MANY CORNERS? (1-4)')
  607 READ(5,*)NHD
      IF(NHD.LT.1.OR.NHD.GT.4)GO TO 607
      CALL HANDI(B,NHD,NAH,OC)
      CALL BOARDP(B,0,0)
      GO TO 100
    7 NAH=OC
      WRITE(6,609)
  606 READ(5,*)NHD
      IF(NHD.LT.1.OR.NHD.GT.4)GO TO 606
      CALL HANDI(B,NHD,NAH,OC)
      CALL BOARDP(B,0,0)
  100 RETURN
      END
      SUBROUTINE HANDI(B,NHD,NAH,OC)
      INTEGER B(10,1)
      INTEGER NCORNI(4),NCORNJ(4)
      DATA NCORNI,NCORNJ /2,2,9,9,2,9,9,2/
      SIGN=-1.
      IF(NAH.EQ.OC)SIGN=+1.
      DO 10 I=1,NHD
      I1=NCORNI(I)
      I2=NCORNJ(I)
   10 B(I1,I2)=SIGN*OC
      RETURN
      END
```

Index to Advertisers

Coming in January

- **How Would You Like a 16-Bit Computer?** Prices of 16-bit microprocessor chips are coming down and some are now second-sourced. Here's a rundown on the *new* features you can expect from 16-bit machines in the near future.
- **How to Write a Computer Simulation.** In simple language, the author shows how to write a computer simulation to be used for classroom instruction in some other subject, using as an example a simulation of power generation, fuel requirements, and costs.
- **Games, Games, Games.** Several new ones you'll want to feed into your computer right away. Complete listings, runs and descriptions, of course.
- **A New Sorting Algorithm.** Using as few comparisons as possible, with a minimum of steps between each comparison, and taking advantage of natural sequencing, this is a modular, easy-to-understand sort program written in BASIC.
- **Grammar as a Programming Language.** This brief but thorough description of Logo shows how a one-command computer language can be highly uncomplicated and very suitable for student tinkering.
- **An 8-Hour Course in BASIC: Part 4.** Readership has been very high for the first three parts of this series, which is an introduction to BASIC by Tom Dwyer, author of "A Guided Tour to Computer Programming in BASIC," one of the top best-sellers on the *Creative Computing* book list.

WE BUILD QUALITY AND USER VERSATILITY INTO EVERY "BLUE BOARD"...

MB3 2K/4K EPROM Board
Kit less EPROM $ 64.95
Kit with 8-1702A (1 μs) 104.95
Kit with 16-1702A (1 μs) 144.95

MB4 4K Memory Board
Kit with 4K 450 ns 100 mw RAM $109.95
Bare P.C. Board 29.95

MB6 8K Memory Board
Kit with 8K 450 ns 100 mw RAM $199.95
Bare P.C. Board 29.95

MB7 16K Memory Board
Kit with 4K RAM $187.95
Kit with 8K RAM 295.00
Kit with 16K RAM 525.00
Bare P.C. Board 29.95

MB8 8K/16K EPROM Board
Kit less EPROM $ 84.95

MB9 Static PROM/RAM Board
Kit less memory $ 79.95

IO2 Input/Output, PROM and Universal Board
Kit $ 54.95
Bare P.C. Board 29.95

VB1 Video Interface Board
Kit $149.95
Bare P.C. Board 29.95

SB1 Music Synthesizer Board
Kit with paper tape of MUS-X1 $199.95

IO4 2-Parallel and 2-Serial Input/Output Board
Kit $149.95

*Assembled boards also available at slightly higher prices.

And, we build just about any board you'll want for S-100 bus expansion.

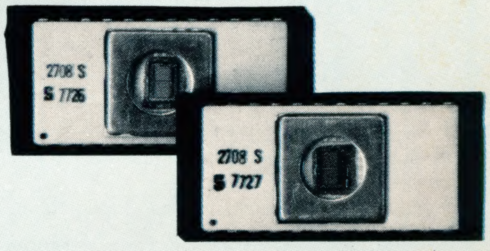
When you're thinking about expansion look to the Solid State Music "blue boards." You'll find quality and user versatility built into every one allowing you to expand your system in whatever direction you choose . . . and, we've been doing it for years.

Right from the start we design our boards with our customers in mind. Extra features are added that will aid in expansion, not hinder program design and development. All first class parts are used and they're checked to make sure you have years of trouble free operation. Plus, every kit comes complete with assembly instructions and user information to make assembly a snap and operation a pleasure.

Talk to your dealer today to get more facts about the "blue boards" or write direct. Compare prices, quality and features. You'll find out why more and more people are using Solid State Music "blue boards" for their S-100 bus expansion.

SPECIAL BONUS OFFER
An 8080 Monitor for 1/2 price!!!

If you buy any of the Solid State Music kits or assembled boards you'll receive a SSM8080 Monitor complete with either eight 1702's or two 2708's and over 50 pages of software information. A $49.95 retail value . . . just $25.00. Hurry!



cybercom

Solid State Music
2102A Walsh Avenue
Santa Clara, CA 95050
(408) 246-2707

We're the blue boards